

# TERRIFIC GAMES FOR THE TI 99/4A

HAL RENKO/SAM EDWARDS



# **Terrific GAMES for the TI99/4A**

Hal Renko and Sam Edwards



**ADDISON-WESLEY PUBLISHING COMPANY**

Reading, Massachusetts • Menlo Park, California

London • Amsterdam • Don Mills, Ontario • Sydney

**© 1983 Addison-Wesley Publishers Limited**

**All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior written permission of the publisher.**

**Cover design by Stuart Hughes  
Illustrations by Agnes Lehár-Graham  
Photos supplied by Barnabys Picture Library**

**Printed in the United States of America.**

**ISBN 0-201-16475-2  
ABCDEFGHIJ-DO-89876543**

# Contents

Zombies in the Swamp*	1
Galactic Monsters	5
Keyboard Memory	10
Las Vegas a Go Go*	13
Parrot	21
Kentucky Derby*	24
Rainbow Square Dance*	28
Qui Vive	32
STM	43
One to Five	46
Escher	50
Genius at Work	55
Shark Hunt	60
Shakespearian Shuffle	65
Explosion*	70
New York, New York*	77
Key	83
Black Box	87
Treasure Hunt	91
I.T. — The Adventure of the Century*	96
The Wolf and the Five Little Goats*	108
Highway*	113
Road Race*	115
At the Market	122
Fallout*	125
Ship's Attack*	129
Mini Mancala*	133
Stop It!	138

<b>BABA</b>	<b>140</b>
<b>Vowels and Consonants</b>	<b>145</b>
<b>Astrology</b>	<b>147</b>

**\* These games require extended BASIC**

## Acknowledgements

This book was born at the University of Twente in the Netherlands where we met the following games fanatics: H. Hermens, P. de Vries, A. Stapel, H. Rieseboos, A. Rensink, M. Sijbenga, E. Groenhuis, W. Koolhoven, R. Bosch, A. Pelsmaeker, K. Boon, and B. Hoogstraten. We would like to thank them all for the many hours they spent with us, discussing, creating, and writing the programs for the games included here.



# Introduction

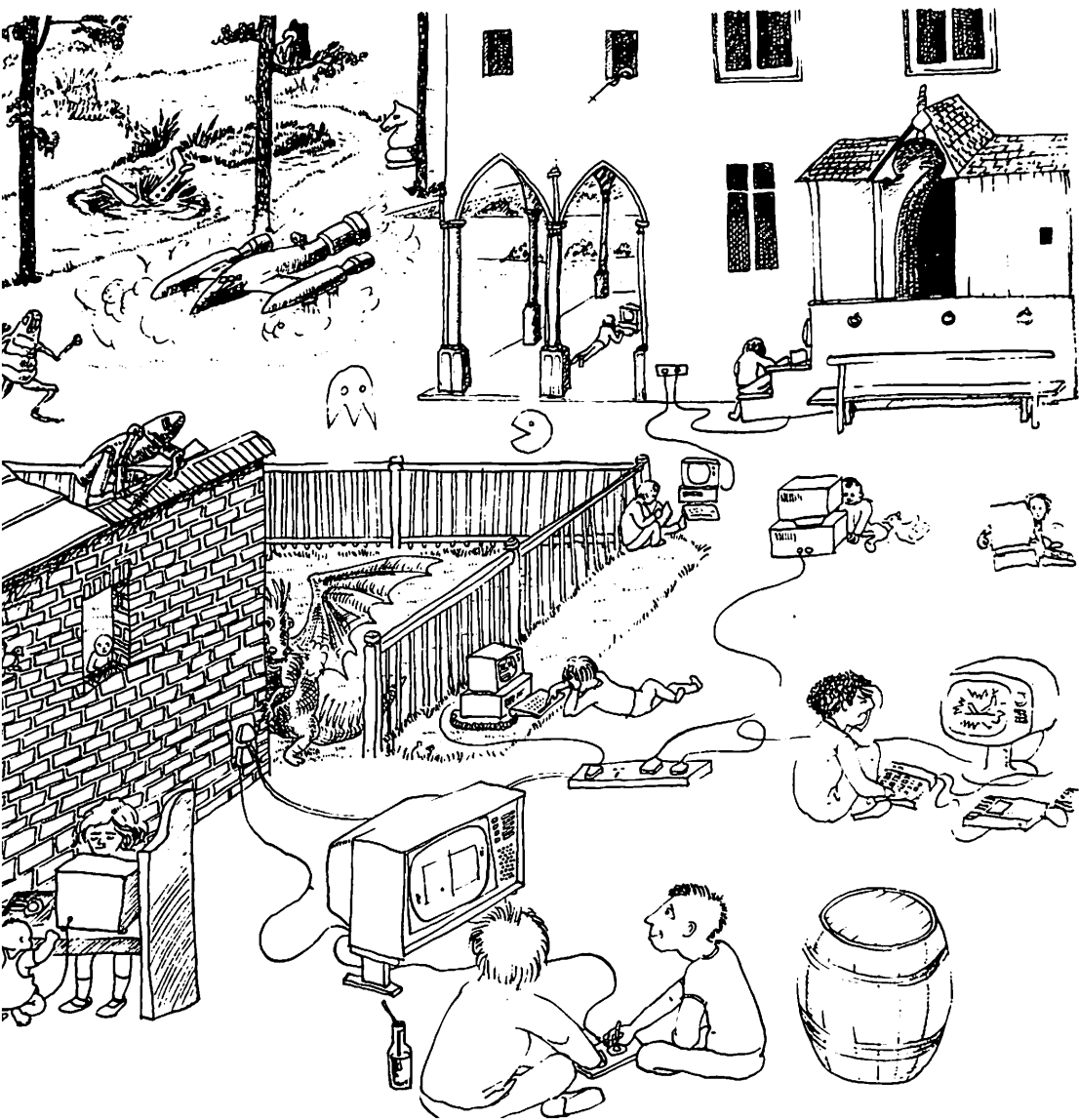
Everybody knows that computers are used for all kinds of serious purposes — financial calculations, business applications and text processing, for instance. They can, however, help us in another way. As the proverb says ‘All work and no play makes Jack a dull boy’ and it’s certainly true that we all need something to keep the cogwheels of our mind turning when we are not considering the ‘important’ things in life. So we read books, do hobbies, and play games.

Bruegel’s wonderful painting ‘Children at Play’ shows us that this has been true for centuries, if not forever.

This book aims to provide something for everybody from 6 to 96. There are games to play on your own, and others to play with your family and friends. You will find fast action games to test your reflexes and your ability to control many moving objects on the screen; puzzles and brainteasers to get you thinking; board games where you pit your wits against the computer; and, of course, some wonderful arcade-type games which combine aspects of all of these. There are also some very intriguing games which are in a class of their own.

A number of the games have been devised to give interesting and worthwhile results from relatively short listings that will not take much time to enter.

Looking again at ‘Children at Play’ a thought springs to mind. If Bruegel were alive today, would his painting look something like this . . .



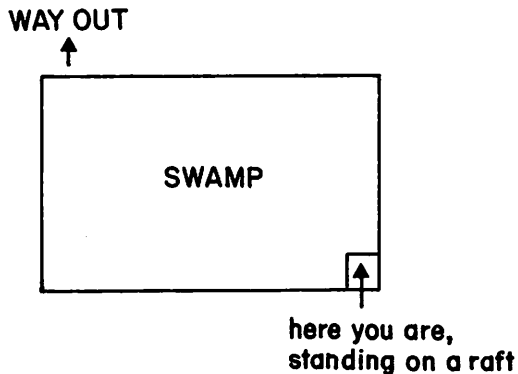




# Zombies in the Swamp

Watch out! There are zombies about, and you have to cross the swamp where they live. To do this you must walk across on rafts.

At the start of the game the swamp looks like this:



You are at one corner, and you must reach the diagonally opposite corner to escape.

The computer asks

MOVE OR THROW? (M OR T)

To put a raft next to the one you're standing on, enter T and the computer asks

## THROW TO THE . . . ?

Enter N for north, S for south, E for east, or W for west and a raft will appear in the appropriate position (north is at the top of the screen). To move enter M, and when the computer asks

## MOVE TO THE . . . ?

again enter the appropriate letter to indicate your direction, but be careful — don't try and step in a direction where you haven't put a raft!

So far so good, but beware! From time to time one of those zombies will emerge from the swamp and run across the rafts you have positioned. If you are in its way it will catch you. Luckily, zombies can't see very well so they often fall off the rafts back into the swamp.

To succeed in this exciting game you must develop your own strategy. The longer and more complicated your path the less likely the zombies are to catch you, but, on the other hand, it will take you longer to get across the swamp, giving time for more zombies to appear.

Oh no! Here come those terrifying zombies again. You had better just hope they don't catch you.

```
10 REM TI(EXTENDED)-ZOMBIES IN THE SWAMP
20 CALL CLEAR
30 DIM MA(11,16,2)
40 RANDOMIZE
50 CALL CHAR(120,"00667E187EB18166")
60 CALL CHAR(130,"18183C5A993C2466")
70 CALL CHAR(140,"FFFFFFFFFFFFFFFF")
80 CALL CHAR(110,"0000001818000000")
90 MA(10,15,1)=1
100 CALL COLOR(0,4,4):: CALL COLOR(12,7,12)
```

```

110 CALL COLOR(13,2,16):: CALL COLOR(14,16,6)
120 CALL COLOR(10,2,4)
130 FOR R=1 TO 10
140 CALL HCHAR(5+R,6,30,15)
150 NEXT R
160 CALL HCHAR(15,20,130)
170 X=10 :: Y=15
180 D$(1)="MOVE OR THROW?(M OR T)"
190 D$(2)="MOVE TO THE ..?"
200 D$(3)="THROW TO THE ..?"
210 DISPLAY AT(20,3)BEEP:D$(1)
220 REM INPUT*****
230 CALL KEY(3,K,S):: IF S=0 THEN 230
240 IF K<>77 AND K<>84 THEN 230
250 ON INT(K/16)-3 GOSUB 300,410
260 TURNS=TURNS+1
270 IF RND<.4 AND TURNS>6 THEN GOSUB 480
280 GOTO 210
290 REM MOVE*****

```



```

300 DISPLAY AT(20,3)BEEP:D$(2)
310 CALL KEY(3,K,S):: IF S=0 THEN 310
320 TX=X-(K=83)+(K=78)
330 TY=Y-(K=69)+(K=87)
340 IF TX=0 AND TY=1 THEN PRINT "You succeeded," :: PRINT "congratulations" :: E
ND
350 IF (TX>10)+(TX<1)+(TY>15)+(TY<1)<>0 THEN 390
360 CALL HCHAR(X+5,Y+5,140):: CALL HCHAR(TX+5,TY+5,130)
370 X=TX :: Y=TY
380 IF MA(X,Y,0)=0 THEN PRINT "you drowned" :: END
390 RETURN
400 REM THROW*****
410 DISPLAY AT(20,3)BEEP:D$(3)
420 CALL KEY(3,K,S):: IF S=0 THEN 420
430 TX=X-(K=83)+(K=78)
440 TY=Y-(K=69)+(K=87)
450 IF (TX>10)+(TX<1)+(TY>15)+(TY<1)=0 THEN CALL HCHAR(TX+5,TY+5,140):: MA(TX,TY
,0)=1
460 RETURN
470 REM MONSTER MOVE*****
480 X1=10 :: Y1=15
490 K=INT(RND*6)
500 TX1=X1+(K=1)-(K=2)+(K=4):: TY1=Y1+(K=3)-(K=0)+(K=5)
510 IF TX1=X AND TY1=Y THEN PRINT "you have been eaten!" :: END
520 IF (TX1>10)+(TX1<1)+(TY1>15)+(TY1<1)<>0 THEN 490
530 IF MA(TX1,TY1,0)=1 THEN CALL HCHAR(X1+5,Y1+5,140):: CALL HCHAR(TX1+5,TY1+5,1
20):: X1=TX1 :: Y1=TY1 :: IF RND<.7 THEN 500 ELSE 490
540 BOL=0
550 FOR C1=-1 TO 1
560 FOR C2=-1 TO 1
570 IF ABS(C1)<>ABS(C2) THEN BOL=BOL+(MA(X1+C1,Y1+C2,1)=1)
580 NEXT C2
590 NEXT C1
600 IF BOL=-3 THEN MA(X1,Y1,1)=1 :: MA(X1,Y1,0)=0
610 IF MA(TX1,TY1,1)=1 THEN 490
620 MA(TX1,TY1,1)=1 :: CALL HCHAR(X1+5,Y1+5,140):: CALL HCHAR(TX1+5,TY1+5,110)
630 RETURN

```

# Galactic Monsters

There it is, registering on the x-y-Gz radar. At last, you are approaching the Milky Way. In just 2.56 protoseconds you will be safe in your home galaxy!

But there is danger here, and as captain of your ship, you know very well what it is. You keep a watchful eye on the XR6-screen. No alert yet. The tension is unbearable . . . 2.5 protoseconds . . . 2.0 protoseconds . . . 1.5 protoseconds . . . oh no! There it is —

## *THE VAN ALLEN SQUARE STONE BELT*

There are a number of these notorious belts, and in them lurk the dreaded Galactic monsters. No weapon can defend you against these vengeful creatures; all you can do is avoid them.

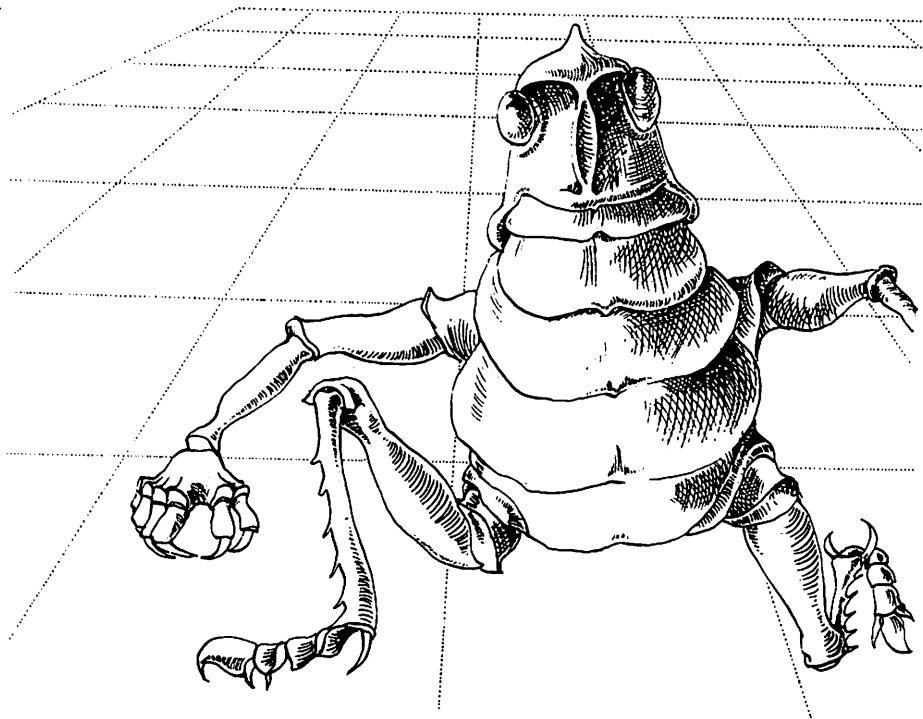
Once the program is started and you see a *VAN ALLEN SQUARE STONE BELT* then you can move yourself by pressing:

- ↑ one square upward
- ← one square to the left
- one square to the right
- ↓ one square downward

Keep on pressing the key until your ship moves (it can take quite some time).

Those terrible Galactic monsters will approach closer and closer. Your only hope for survival is to avoid them! If you cross a *VAN ALLEN SQUARE STONE BELT* a second time, the monsters will become more aggressive. Somehow, they seem to anticipate your every move. If they manage to attack you five times, you and your ship are totally destroyed.

Good luck, captain . . . only a few protoseconds to go, and you are home free!



```

10 REM TI-GALACTIC MONSTERS
20 A1$="0066421818423C00"
30 A2$="007E7E7E7E7E00"
40 A3$="18183C5A993C2466"
50 CALL COLOR(1,8,16)
60 CALL COLOR(2,9,16)
70 CALL COLOR(3,2,16)
80 CALL CHAR(35,A1$)
90 CALL CHAR(45,A2$)

```

```

100 CALL CHAR(55,A3$)
110 DIM X(12)
120 DIM Y(12)
130 GOTO 1070
140 REM SCREEN*****
150 CALL CLEAR
160 FOR T=1 TO 11
170 CALL HCHAR(5+T,10,45,11)
180 NEXT T
190 CALL SOUND(300,262,0)
200 CALL SOUND(300,330,0)
210 CALL SOUND(300,392,0)
220 CALL SOUND(1000,262,0,330,0,392,0)
230 CALL HCHAR(13,10,35,11)
240 CALL HCHAR(6,15,55)
250 X1=6
260 Y1=15
270 FOR T=1 TO 11
280 X(T)=13
290 Y(T)=9+T
300 NEXT T
310 RETURN
320 REM PLAYER'S MOVE*****
330 CALL SOUND(10,880,0)
340 CALL KEY(3,KEY,ST)
350 CALL HCHAR(X1,Y1,45)
360 TURNS=TURNS+1
370 IF KEY=88 THEN 420
380 IF KEY=83 THEN 450
390 IF KEY=68 THEN 480
400 IF KEY=69 THEN 510
410 GOTO 530
420 IF X1=16 THEN 530
430 X1=X1+1
440 GOTO 530
450 IF Y1=10 THEN 530
460 Y1=Y1-1
470 GOTO 530
480 IF Y1=20 THEN 530
490 Y1=Y1+1
500 GOTO 530

```



```

510 IF X1=6 THEN 530
520 X1=X1-1
530 CALL HCHAR(X1,Y1,55)
540 RETURN
550 REM CAUGHT*****
560 FOR T=1 TO 11
570 IF X1<>X(T) THEN 640
580 IF Y1<>Y(T) THEN 640
590 CALL SOUND(-100,440,0)
600 CALL SOUND(-100,220,0)
610 CALL SOUND(-100,110,0)
620 CALL SOUND(1000,-4,0)
630 K=K+1
640 NEXT T
650 RETURN
660 REM MONSTER'S MOVE***
670 FOR T=1 TO 11
680 CALL HCHAR(X(T),Y(T),45)
690 IF RND<CHANCE THEN 950
700 A=INT(RND*4)
710 IF A=0 THEN 750
720 IF A=1 THEN 770
730 IF A=2 THEN 790
740 GOTO 810
750 X(T)=X(T)+1
760 GOTO 820
770 X(T)=X(T)-1
780 GOTO 820
790 Y(T)=Y(T)+1
800 GOTO 820
810 Y(T)=Y(T)-1
820 IF X(T)<6 THEN 870
830 IF X(T)>16 THEN 890
840 IF Y(T)<10 THEN 910
850 IF Y(T)>20 THEN 930
860 GOTO 940
870 X(T)=X(T)+1
880 GOTO 1010
890 X(T)=X(T)-1
900 GOTO 1010
910 Y(T)=Y(T)+1

```

```

920 GOTO 1010
930 Y(T)=Y(T)-1
940 GOTO 1010
950 X2=X1-X(T)
960 Y2=Y1-Y(T)
970 IF RND<.8 THEN 1000
980 X(T)=X(T)+SGN(X2)
990 GOTO 1010
1000 Y(T)=Y(T)+SGN(Y2)
1010 CALL HCHAR(X(T),Y(T),35)
1020 NEXT T
1030 FOR U=1 TO 11
1040 CALL HCHAR(X(U),Y(U),35)
1050 NEXT U
1060 RETURN
1070 REM main program*****
1080 CHANCE=CHANCE+.1
1090 GOSUB 140
1100 GOSUB 320
1110 GOSUB 550
1120 GOSUB 660
1130 IF K>4 THEN 1220
1140 IF X1<16 THEN 1100
1150 SCORE=SCORE+POINTS-TURNS-5*K
1160 POINTS=POINTS+20
1170 TURNS=0
1180 GOTO 1080
1190 CALL SOUND(-100,110,0)
1200 CALL SOUND(1000,-4,0)
1210 RETURN
1220 PRINT "End of this game."
1230 PRINT "Your score is:"
1240 PRINT SCORE
1250 PRINT "For another game:enter'run'."
1260 END

```

# Keyboard Memory

This game uses the computer keyboard to test your memory. Eighteen of the keys each conceal a digit, in the same way that a playing card, lying face down, hides its value. There are nine different digits, each hidden by two keys.

The symbols of the keys you must concentrate on are shown on the screen.

Q	W	E	R	T	Y
A	S	D	F	G	H
Z	X	C	V	B	N

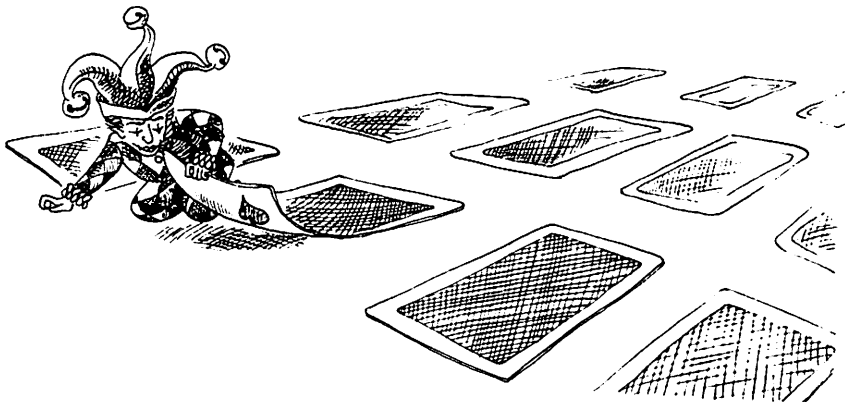
Press a key and the digit it is hiding will be shown on the screen: press two, one after the other, and if they are both hiding the same digit their symbols will disappear from the screen.

Your first few attempts will obviously be guesses but by memorizing the positions of the digits they show, you should soon be able to work out where the pairs are. How few turns will it take you to clear the screen completely?

```

10 REM TI-KEYBOARD MEMORY
20 RANDOMIZE
30 R=INT(RND*9)+1
40 CALL CLEAR
50 DIM T$(18),W$(18)
60 A$="QWERTYASDFGHZXCVBN"
70 B$="194338541967725268"
80 GOTO 550
90 REM COMPOSE RANDOM LAYOUT*****
100 FOR N=1 TO 18
110 T$(N)=SEG$(A$,N,1)
120 DIS=N+R
130 IF DIS<19 THEN 150
140 DIS=DIS-18
150 W$(N)=SEG$(B$,DIS,1)
160 NEXT N
170 RETURN
180 REM SCREEN*****
190 FOR K=1 TO 3
200 REM TURN CARD*****
210 FOR J=1 TO 6
220 CALL HCHAR(4*K,4*J,ASC(T$((K-1)*6+J)))

```



```

230 NEXT J
240 NEXT K
250 RETURN
260 REM INPUT*****
270 Q$(2)=" "
280 FOR KK=1 TO 2
290 CALL KEY(3,KE,STAT)
300 IF STAT=0 THEN 290
310 Q$(KK)=CHR$(KE)
320 IF Q$(1)=Q$(2) THEN 290
330 STAT=0
340 NN(KK)=0
350 FOR N=1 TO 18
360 K=INT((N-0.5)/6)+1
370 J=N-(K-1)*6
380 IF T$(N)<>Q$(KK) THEN 420
390 TU=TU+1
400 CALL HCHAR(4*K,4*J,ASC(W$(N)))
410 NN(KK)=N
420 NEXT N
430 IF NN(KK)=0 THEN 290
440 NEXT KK
450 RETURN
460 REM CHECK CARDS*****
470 IF W$(NN(1))<>W$(NN(2)) THEN 530
480 TEL=TEL+1
490 T$(NN(1))=" "
500 T$(NN(2))=" "
510 RETURN
520 IF TEL=9 THEN 540
530 GOTO 190
540 PRINT "NUMBER OF TURNS=";TU
550 REM MAIN PROGRAM*****
560 GOSUB 100
570 GOSUB 190
580 GOSUB 280
590 GOSUB 470
600 IF TEL<9 THEN 570
610 PRINT "NUMBER OF TURNS=";TU
620 END

```

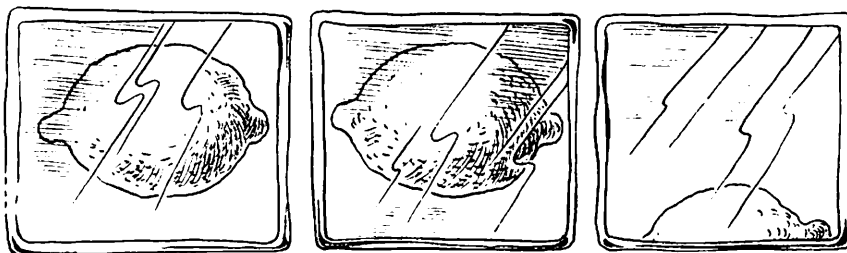
# Las Vegas a Go Go

Have you ever watched someone pumping coin after coin into a one-armed bandit, and found yourself wondering what the fascination was? You will soon find out when this program turns your computer into a fabulous Las Vegas-style fruit machine. All the playing instructions you need will appear on the screen. At certain points you will be presented with a list of options, for instance

INSERT, HOLD, PLAY OR END

Enter your choice by typing the first letter of the option you want, for instance P keeps your machine playing. The reels are numbered 1, 2, and 3. If you want to hold one or more reels, type in the appropriate number or numbers after you have entered H.

Lights will flash and music play as the wheels whiz around. Have you won this time? Never mind, you are sure to hit the jackpot sooner . . . or later!



```

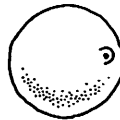
10 REM TI(EXTENDED)-LAS VEGAS A GOGO
20 CALL CLEAR :: RANDOMIZE
30 DIM WF(13),R(2,13),JAC(13):: JAC(5)=-1 :: JAC(6)=-1 :: JAC(7)=-1
40 FOR VV=1 TO 13 :: READ WF(VV),R(1,VV),R(2,VV):: NEXT VV
50 DATA 110,0,0,116,0,0,115,0,0,109,0,0,106,8,50,105,8,40,114,7,30
60 DATA 113,5,25,104,3,20,136,3,20,137,2,10,128,2,10,112,0,0
70 DIM JPX(4),JPY(4),JF(4),JCC(4):: FOR JO=1 TO 4 :: READ JPX(JO),JPY(JO),JF(JO)
:: NEXT JO
80 DATA 11,8,110,12,8,116,14,8,115,15,8,109
90 NJ%=CHR$(138)
100 DIM WPX(3),WPY(3),V(4),HV(3):: WPX(1)=11 :: WPX(2)=13 :: WPX(3)=15 :: FOR I
=1 TO 3 :: WPY(I)=10 :: NEXT I
110 DIM HPX(3),HPY(3),HF(3):: FOR I=1 TO 3 :: HPX(I)=WPX(I):: HPY(I)=13 :: NEXT
I
120 H1%=CHR$(107):: NH%=CHR$(108):: HH%=CHR$(96):: HO%=H1%&HH%&H1%&HH%&H1%
130 DIM GPX(4),GPY(4),GV(4):: GPX(1)=22 :: GPY(1)=1
140 GPX(2)=27 :: GPY(2)=2 :: GPX(3)=22 :: GPY(3)=3 :: GPX(4)=17 :: GPY(4)=2
150 NJ=4 :: NH=0 :: NG=0 :: GW=0 :: NI=0 :: NT=-1
160 GOTO 1740
170 REM BLINK S1$/S2$ AND GET IN
180 DISPLAY AT(PLY,PLX):S1$:: CH=0 :: FOR DE=1 TO 10
190 IF CH=0 THEN CALL KEY(0,GET,CH)
200 NEXT DE :: DISPLAY AT(PLY,PLX):S2$:: IF CH=1 THEN RETURN
210 FOR DE=1 TO 10 :: IF CH=0 THEN CALL KEY(0,GET,CH)
220 NEXT DE :: IF CH=1 THEN RETURN
230 GOTO 180
240 REM ADD INCREMENT TO MONEY
250 FOR AD=SGN(IN) TO IN STEP SGN(IN)
260 DISPLAY AT(1,7):M0+AD:
270 IF SGN(IN)>0 THEN 290
280 CALL SOUND(-50,200,2):: GOTO 300
290 FOR S0=500 TO 700 STEP 100 :: CALL SOUND(-50,S0,2):: NEXT S0
300 NEXT AD :: M0=M0+IN :: RETURN
310 REM REMOVE DOUBLE
320 DO=0 :: CALL HCHAR(2,3,32,7)
330 FOR JO=1 TO 4 :: CALL HCHAR(JPY(JO),JPX(JO)+2,JF(JO)):: JCC(JO)=0 :: NEXT JO
340 NJ=4 :: RETURN
350 CALL HCHAR(24,3,32,28):: RETURN
360 REM JACKPOT
370 FOR TI=1 TO 4 :: CALL HCHAR(4,3,32,JC)
380 FOR C=1 TO JC :: CALL SOUND(-100,150+20*C,0):: DISPLAY AT(4,C):"J":: NEXT C

```

```

390 NEXT TI :: RETURN
400 REM DEFINE CHARACTERS
410 CALL CHAR(96,"FFFFFFFFFFFFFF")
420 CH$="0F0F0F0F0F0F0F0F" :: CALL CHAR(97,CH$)
430 CALL CHAR(60,CH$)
440 CALL COLOR(9,6,1,10,2,16,11,7,16,12,16,6,13,11,16,14,13,16)
450 CALL CHAR(107,"FFBBBBBB3BBBBBB")
460 CALL CHAR(108,"FFFFFFFFFFFFFF")
470 A$="1898FF3D3C3CE404"
480 CALL CHAR(109,A$):: CALL CHAR(115,A$)
490 A$="1819FFBC3C3C2720"
500 CALL CHAR(110,A$):: CALL CHAR(116,A$)
510 CALL CHAR(136,"02043C566A566A3C")
520 CALL CHAR(114,"00006C7C7C3B1000")
530 CALL CHAR(112,"02043B7C7C7C3B00")
540 CALL CHAR(104,"02020C3C7B7B3000")
550 CALL CHAR(128,"00406060703C1E00")
560 CALL CHAR(137,"02041C3C7C7C3B00")
570 CALL CHARPAT(36,A$):: CALL CHAR(106,A$)
580 CALL CHARPAT(63,A$):: CALL CHAR(120,A$)
590 CALL CHAR(113,"0B1C3E7F3E1C0B00")
600 CALL CHAR(104,"0B1C2A772A0B0B00")
610 CALL CHAR(105,"00103B7C7C103B00")

```

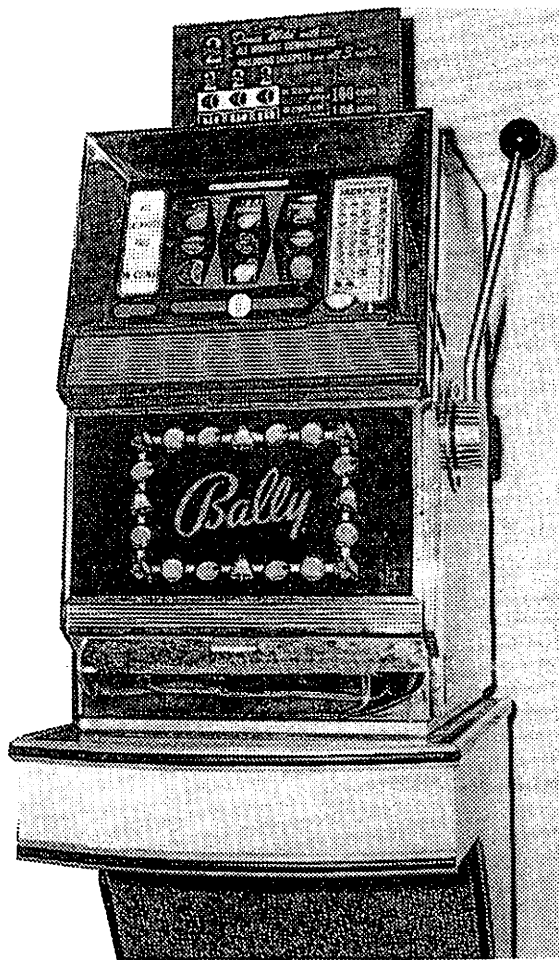


```

620 CALL CHAR(138,"0000000000000000")
630 RETURN
640 REM ** DRAW SCREEN
650 CALL CLEAR :: CALL SCREEN(12)
660 CALL HCHAR(7,12,96,7)
670 FOR I=8 TO 12 :: CALL HCHAR(I,11,96,9):: NEXT I
680 CALL VCHAR(7,20,60)
690 CALL VCHAR(8,20,97,4)

```





```

700 CALL HCHAR(12,20,96)
710 CALL HCHAR(13,12,96,7)
720 FOR I=14 TO 19 :: CALL HCHAR(I,13,96,5):: NEXT I
730 CALL HCHAR(20,12,96,7)
740 CALL HCHAR(21,11,96,9)
750 CALL HCHAR(22,10,96,11)
760 DISPLAY AT(1,1):"TOTAL: 0"
770 DISPLAY AT(10,3):CHR$(112);" = 1";
780 FOR VV=12 TO 5 STEP -1
790 DISPLAY AT(23-VV,1):"-";RPT$(CHR$(WF(VV)),2);"-";R(1,VV);
800 DISPLAY AT(23-VV,20):RPT$(CHR$(WF(VV)),3);"=";R(2,VV);
810 IF JAC(VV)THEN DISPLAY AT(23-VV,27):"+J";
820 NEXT VV :: RETURN
830 REM ** ADAPT VARIABLES
840 NT=NT+1 :: IF WI>0 THEN HB=-1 :: WI=0
850 FOR HO=1 TO 3 :: HF$(HO)=NH$ :: DISPLAY AT(HPY(HO),HPX(HO)):NH$:: NEXT HO
860 IF DO THEN 910
870 FOR JO=1 TO 4 :: IF JCC(JO)=0 THEN 900
880 JCC(JO)=JCC(JO)-1 :: IF JCC(JO)>0 THEN 900
890 NJ=NJ+1 :: CALL HCHAR(JPY(JO),JPX(JO)+2,JF(JO))
900 NEXT JO :: GOTO 920
910 DC=DC-1 :: IF DC=0 THEN GOSUB 320
920 IF JA THEN DISPLAY AT(4,JC):" ";: JC=JC-1 :: IF JC=0 THEN JA=0
930 RETURN
940 REM ** NOT ENOUGH MONEY
950 HB=-1 :: GOSUB 320
960 JA=0 :: CALL HCHAR(4,3,32,15)
970 GOSUB 350 :: DISPLAY AT(24,1):"INSERT OR END";
980 S1$="INSERT (I)" :: S2$=RPT$(" ",10):: PLX=1 :: PLY=2 :: GOSUB 180
990 IF GET=ASC("P")THEN 980 ELSE RETURN
1000 REM ** HOLD POSSIBLE
1010 DISPLAY AT(24,1):"INSERT, HOLD, PLAY OR END";
1020 S1$=HO$ :: S2$=HF$(1)&HH$&HF$(2)&HH$&HF$(3)
1030 PLX=11 :: PLY=13 :: GOSUB 180
1040 IF GET<49 OR GET>51 THEN RETURN ELSE HO=GET-48
1050 NH=NH+1 :: IF HF$(HO)=NH$ THEN HF$(HO)=H1$ ELSE HF$(HO)=NH$
1060 GOTO 1020
1070 REM ** NO HOLD
1080 GOSUB 350 :: DISPLAY AT(24,1):"INSERT, PLAY OR END";
1090 S1$=RPT$(CHR$(120),3):: S2$=RPT$(CHR$(96),3):: PLX=12 :: PLY=19 :: GOSUB 18
0

```

```

1100 RETURN
1110 REM ** WHAT TO DO WITH WINNINGS?
1120 GOSUB 350 :: IF NOT HB THEN DISPLAY AT(24,1):"HOLD, ";
1130 DISPLAY AT(24,7):"GAMBLE OR COLLECT";
1140 GOTO 1090
1150 REM ** SPIN GAMBLE WHEELS
1160 FOR I=7 TO 10 :: CALL HCHAR(I,20,32):: CALL HCHAR(I+1,20,60):: NEXT I
1170 CALL SOUND(-4000,-7,29)
1180 FOR I=10 TO 7 STEP -1 :: CALL HCHAR(I,20,60):: CALL HCHAR(I+1,20,97):: NEXT
    I
1190 FOR WD=1 TO 3 :: IF HF$(WD)=NH$ THEN DISPLAY AT(WPY(WD),WPX(WD)):CHR$(13B);
1200 NEXT WD
1210 FOR WD=1 TO 3 :: IF HF$(WD)=H1$ THEN 1240
1220 FI=INT(RND*100+1):: IF FI<5 THEN IF JCC(FI)<>0 THEN V(WD)=13 ELSE V(WD)=FI
    :: JCC(FI)=-1 :: GOTO 1240
1230 V(WD)=5-(FI>7)-(FI>10)-(FI>13)-(FI>23)-(FI>36)-(FI>49)-(FI>68)-(FI>87)
1240 FOR DE=2 TO 300 :: NEXT DE :: DISPLAY AT(WPY(WD),WPX(WD)):CHR$(13B);

```



```

1250 CALL SOUND(-100,300,2):: CALL HCHAR(WPY(WD),WPX(WD)+2,WF(V(WD)))
1260 CALL SOUND(4000,-7,29):: NEXT WD :: CALL SOUND(-1,-2,30):: RETURN
1270 REM ** TAKE CARE OF JOKERS
1280 J=0 :: FOR WD=1 TO 3 :: VV=V(WD)
1290 IF VV>4 THEN FV=VV :: GOTO 1320
1300 JW=WD :: J=J+1 :: IF JCC(VV)>0 THEN 1320
1310 DISPLAY AT(JPY(VV),JPX(VV)):NJ$:: JCC(VV)=20 :: NJ=NJ-1
1320 NEXT WD :: DO=(NJ=0):: IF DO THEN DC=15 :: DISPLAY AT(2,1):"DOUBLE!";
1330 RETURN
1340 REM ** COMPUTE WINNINGS
1350 HV=13 :: ON J+1 GOTO 1360,1380,1360,1410
1360 IF NOT(JA AND FV>10) THEN HV=FV
1370 GOTO 1410
1380 V(0)=V(3):: V(4)=V(1):: IF JA AND V(JW-1)<>V(JW+1) THEN 1410
1390 V(0)=15 :: V(4)=15

```

```

1400 IF V(JW+1)>V(JW-1)THEN HV=V(JW-1)ELSE HV=V(JW+1)
1410 FOR WD=1 TO 3 :: IF V(WD)<5 THEN HVV(WD)=HV ELSE HVV(WD)=V(WD)
1420 NEXT WD :: IF HVV(1)<>HVV(2)OR HVV(2)<>HVV(3)THEN 1440
1430 IF JAC(HVV(1))THEN JA=-1 :: JC=15 :: GOSUB 370
1440 FOR WD=1 TO 3 :: IF HVV(WD)=13 THEN WI=WI+1
1450 NEXT WD :: IF JA THEN WI=10*WI
1460 NS=-(HVV(1)=HVV(2))-(HVV(2)=HVV(3)):: IF NS>0 THEN WI=WI+R(NS,HVV(2))
1470 IF DO THEN WI=2*WI
1480 RETURN
1490 REM ** GAMBLE ROUTINE
1500 DT=1 :: GV(1)=2*WI :: GV(2)=0 :: GV(3)=INT(3*WI/2):: GV(4)=INT(WI/2)
1510 KEY=0 :: NG=NG+1 :: GOSUB 350 :: DISPLAY AT(24,1):"STOP";
1520 FOR LI=1 TO 3 :: CALL HCHAR(LI,16,96,17):: NEXT LI
1530 RR=RR+1 :: IF RR>4 THEN RR=1
1540 DISPLAY AT(GPY(RR),GPX(RR)):STR$(GV(RR));
1550 CALL SOUND(-4000,150+50*RR,2)
1560 IF KEY<>ASC("S")THEN CALL KEY(0,KEY,CH):: GOTO 1580
1570 DT=(1+RND*2)*DT :: FOR DE=1 TO DT :: NEXT DE :: IF DT>150 THEN 1590
1580 CALL HCHAR(GPY(RR),GPX(RR)+2,96,4):: GOTO 1530
1590 FOR LI=1 TO 3 :: CALL HCHAR(LI,16,32,17):: NEXT LI
1600 GW=GW+GV(RR)-WI :: WI=GV(RR):: CALL SOUND(-1,150+50*RR,2):: RETURN
1610 REM ** END OF GAME
1620 CALL CLEAR :: CALL CHARSET :: CALL SCREEN(8)
1630 DISPLAY AT(5,1):"AMOUNT OF MONEY:"
1640 IMAGE " $ ####.##"
1650 DISPLAY AT(7,3):"PUT IN:":: DISPLAY AT(7,18):USING 1640:NI
1660 DISPLAY AT(8,3):"GOT BACK:":: DISPLAY AT(8,18):USING 1640:M0/4
1670 DISPLAY AT(9,3):"MAX AT ONE TIME:":: DISPLAY AT(9,18):USING 1640:MM/4
1680 DISPLAY AT(10,3):"WON BY GAMBLING:":: DISPLAY AT(10,18):USING 1640:GW/4
1690 DISPLAY AT(12,1):"NUMBER OF HOLDS: ";NH
1700 DISPLAY AT(13,1):"NUMBER OF GAMBLES:";NG
1710 DISPLAY AT(14,1):"NUMBER OF TURNS: ";NT
1720 RETURN
1730 REM MAIN PROGRAM
1740 GOSUB 410 !DEF CHAR
1750 GOSUB 650 !SCREEN
1760 GOSUB 840 !ADAPT
1770 ON -2*HB-(M0>1)+1 GOSUB 950,1010,950,1080
1780 DISPLAY AT(5,9):: CA=-(GET=ASC("I"))-2*(GET=ASC("P"))-3*(GET=ASC("E"))
1790 ON CA+1 GOTO 1770,1800,1820,2020
1800 NI=NI+1 :: IN=4 :: GOSUB 250

```

```

1810 GOTO 1770
1820 IN=-2 :: GOSUB 250
1830 GOSUB 1160 !SPIN WHEELS
1840 IF D0 THEN 1860
1850 GOSUB 1280 !JOKERS
1860 GOSUB 1350 !WINNINGS
1870 IF HF$(1)=H1$ OR HF$(2)=H1$ OR HF$(3)=H1$ THEN 1890
1880 HB=0 :: LW=WI :: IF WI>0 THEN 1910 ELSE 1760
1890 HB=-1 :: IF WI>LW THEN 1910
1900 DISPLAY AT(5,9):" YOU LOST" :: CALL SOUND(-600,200,2):: GOTO 1760
1910 DISPLAY AT(5,9):"YOU WON";: DISPLAY AT(5,17):WI;: FOR DU=1 TO WI :: CALL
SOUND(-50,500,1):: CALL SOUND(1,500,30):: NEXT DU
1920 IF M0<2 THEN HB=-1
1930 GOSUB 1120 !GET INSTRUCTION
1940 CA=-(GET=ASC("H")AND NOT HB)-2*(GET=ASC("G"))-3*(GET=ASC("C"))
1950 ON CA+1 GOTO 1930,2010,1990,1960
1960 IN=WI :: GOSUB 250
1970 IF WI>MM THEN MM=WI
1980 GOTO 1760
1990 HB=-1 :: GOSUB 1500 !GAMBLE
2000 IF WI>0 THEN 1910 ELSE 1900
2010 WI=0 :: GOTO 1760
2020 GOSUB 1620 !END
2030 END

```

# Parrot

PARROT is a very simple competition game. Play it on a rainy day with a friend (especially after your friend has had a few beers!).

At the start of the game, your computer impertinently asks you:



## ARE YOU READY PARROT?

Yes, he is calling you a parrot! Press any key and the computer immediately displays a letter. You must enter the same letter as fast as you can. Since you copy each letter, you really are a parrot. The game is repeated for about 10 seconds, and then the computer shows your score.

How good are you as a parrot? Have a competition with your friends and try to win the title of 'Super-Duper Parrot'. (My record is 19 — I teach parrots!)

```
10 REM TI-PARROT
20 CALL CHAR(50,"1C3A3F3C3C3C7C7E")
30 CALL CHAR(51,"7EFEFE7C101000")
40 CALL COLOR(3,7,1)
50 P=0
60 RANDOMIZE
70 DATA BAD,BAD,NOT SO GOOD,NOT SO GOOD,TRY AGAIN,TRY AGAIN,THAT'S BETTER,THAT'S
  BETTER
80 DATA YOU ARE LEARNING,YOU ARE LEARNING,GOOD,GOOD,VERY GOOD,VERY GOOD,EVEN BET
  TER,EVEN BETTER
90 DATA ONE OF THE BEST,ONE OF THE BEST,YOU'VE PLAYED ENOUGH WITH THIS PROGRAM,Y
  OU'RE THE TOP OF THE BILL
100 CALL CLEAR
110 PRINT "ARE YOU READY PARROT?"
120 CALL KEY(3,K,S)
130 IF S=0 THEN 120
140 CALL CLEAR
150 CALL HCHAR(5,15,50)
160 CALL HCHAR(6,15,51)
170 D=INT(RND*26)+97
180 P=P+1
190 IF P=20 THEN 280
200 CALL HCHAR(10,P+1,D)
210 CALL KEY(5,I,S)
220 IF S=0 THEN 250
230 IF I=D THEN 170
240 IF I=D-32 THEN 170
```

```
250 B=B+1
260 IF B=150 THEN 280
270 GOTO 210
280 PRINT "END"
290 FOR T=1 TO P
300 READ A$
310 NEXT T
320 PRINT A$
```



# Kentucky Derby

Ladies and gentlemen, they are off! Red Arrow is off to a good start . . . Blondish Boy is giving his jockey some trouble . . . and there's the famous Spanish Lady, on the inside track. This is really first class excitement, and we've only just begun! Coming round the first bend . . . Mickey Finn has taken over the lead, hard-pressed by Speedy Gonzales . . .

The Kentucky Derby is a horse-racing game. There are three runners, and the race is two laps long.

At the start of the game the odds are given for each horse. Now you can place your bets. As long as you have money you can bet again and again — these horses never get tired.

After the RUN command we see:

KENTUCKY DERBY

and

YOU HAVE . . . \$

ON WHICH HORSE DO YOU WANT TO BET?

THE ODDS ARE

HORSE 1: 5 to 1

HORSE 2: 4 to 1

HORSE 3: 3 to 1

Now you enter your choice, and the computer will ask you how much you want to bet. When you have run out of money, your computer cheekily tells you:

YOU'RE LIVING BEYOND YOUR MEANS

Immediately after you have entered all your bets, the screen displays the racetrack, with the horses at the starting gates, and off they go!

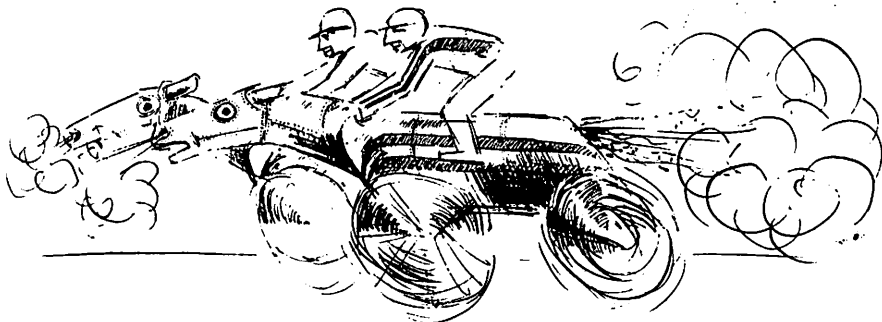
*I put my money on the bobtailed nag  
Somebody bet on the bay*

```
10 REM TI(EXTENDED)-THE KENTUCKY DERBY
20 RANDOMIZE
30 CALL CLEAR
40 PRINT "KENTUCKY DERBY"
50 PRINT ""
60 PRINT ""
70 CREDIT=100
80 REM MAKE BETS*****
90 SC=0
100 PRINT "YOU HAVE";CREDIT;"$"
110 IF CREDIT=0 THEN 410
120 PRINT "ON WHICH HORSE DO YOU WANT"
130 PRINT "TO BET?"
140 TR=0
150 PRINT "THE ODDS ARE:"
160 FOR A=1 TO 3
170 RA(A)=INT(RND*5+1)
180 PRINT "    HORSE";A;" ";RA(A);" TO 1"
190 NEXT A
200 PRINT
210 INPUT PNR
220 IF (PNR<1)+(PNR>3)<>0 THEN 210
230 INPUT "HOW MUCH DO YOU SPEND: ":HM
240 IF HM<=CREDIT THEN 280
250 PRINT "YOU'RE LIVING BEYOND YOUR MEANS"
260 GOTO 230
270 REM HORSE RACE*****
280 GOSUB 430
290 GOSUB 670
300 PW=-((PP(1)>=PP(2)AND PP(1)>=PP(3))+2*(PP(2)>=PP(3)AND PP(2)>PP(1))+3*(PP(3)
>PP(1)AND PP(3)>PP(2)))
310 IF TR<4 THEN 290
320 CALL DELSPRITE(ALL)
```

```

330 PRINT "THE WINNING HORSE IS ";PW
340 IF PW<>PNR THEN 380
350 CREDIT=CREDIT+(RA(PW)-1)*HM
360 PRINT "YOU WON"
370 GOTO 80
380 PRINT "YOU LOST"
390 CREDIT=CREDIT-HM
400 GOTO 80
410 PRINT "YOU'RE LIVING ABOVE YOUR    MEANS"
420 END
430 REM INITIALIZE HORSES**
440 CALL CLEAR
450 CALL CHAR(60,"00040EFF785C8200")
460 CALL CHAR(61,"000C1E383E783010")
470 CALL CHAR(88,"AA55AA55AA55AA55")
480 CALL CHAR(62,"0101010101010101")
490 MUL=1/RA(1)+1/RA(2)+1/RA(3)
500 MUL=1/MUL
510 CH=RND
520 FOR A=1 TO 3
530 J(A)=0
540 VP(A)=2.7+SGN(RND-0.5)/10
550 IF CH<=0 THEN W(A)=0 :: GOTO 580

```



```

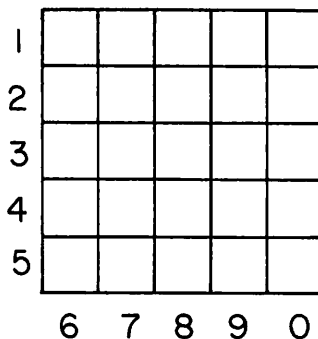
560 W(A)=CH<(MUL/RA(A))
570 CH=CH-(MUL/RA(A))
580 CALL SPRITE(#A,60,2,48+24*A,32)
590 CALL SPRITE(#A+6,88,3,48+24*A,120)
600 CALL SPRITE(#A+3,62,2,48+24*A,230)
610 NEXT A
620 REM START HORSES
630 FOR A=1 TO 3
640 CALL MOTION(#A,0,VP(A))
650 NEXT A
660 RETURN
670 REM MOVE ONE STEP*****
680 SC=SC+1
690 FOR A=1 TO 3
700 IF SC>=30 AND W(A)AND PW<>A THEN VP(A)=VP(A)+.5 :: GOTO 720
710 VP(A)=VP(A)+SGN(RND-0.5)/10+(VP(A)>3.0)-(VP(A)<2.5)
720 CALL POSITION(#A,X,PP(A))
730 HV=0
740 IF PP(A)<95 OR PP(A)>=130 THEN 750 ELSE 780
750 IF X>(48+24*A)THEN HV=-1
760 IF X<(48+24*A)THEN HV=1
770 GOTO 800
780 IF PP(A)>112 THEN CALL SPRITE(#A,60,2,X,PP(A),VP(A),VP(A))ELSE CALL SPRITE(
A,61,2,X,PP(A),-VP(A),VP(A))
790 GOTO 810
800 CALL MOTION(#A,HV,VP(A))
810 F(A)=0
820 IF J(A)THEN 840 ELSE CALL COINC(#A,#A+6,50,J(A))
830 GOTO 860
840 CALL COINC(#A,#A+3,10,F(A))
850 IF F(A)THEN TR=TR+1 :: J(A)=0
860 NEXT A
870 FOR A=1 TO 3
880 CALL POSITION(#A,X,PP(A))
890 NEXT A
900 RETURN

```

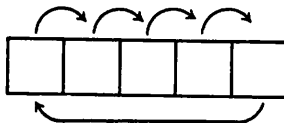
# Rainbow Square Dance

A two-dimensional cube? Impossible of course, but this game, played on the computer screen, is similar in many ways to Rubik's Cube.

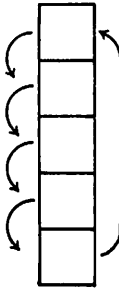
You will see 25 colored squares arranged at random on a  $5 \times 5$  board. The rows of squares on the board are numbered like this:



That is, the horizontal rows are numbered 1 to 5 and the vertical rows 6, 7, 8, 9, and 0. When you enter the number of a row, the squares in that row are moved one position. In a horizontal row the squares move to the right, and in a vertical row they move downwards. As a square disappears off one end of a row it reappears at the other end, like this



in a horizontal row, or this



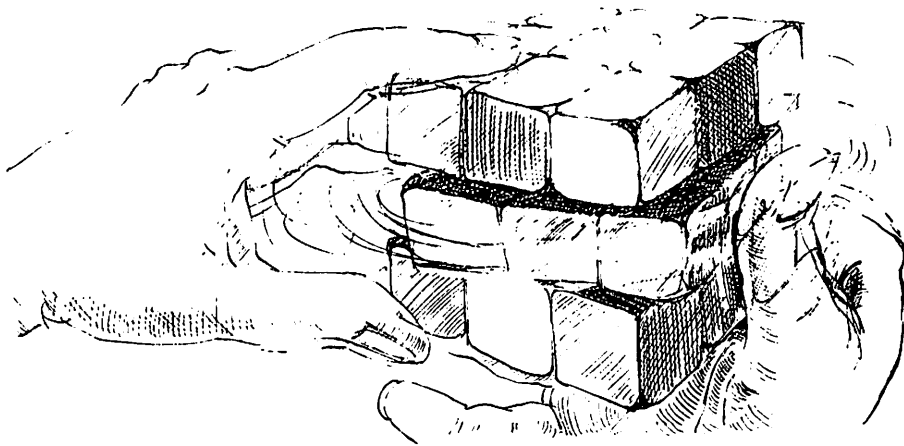
in a vertical row.

The object of all this? You must rearrange the squares to form five horizontal stripes of one color each. Sounds simple . . . but is it?

```

10 REM TI(EXTENDED)-RAINBOW SQUARE DANCE
20 CALL CLEAR
30 RANDOMIZE
40 GOTO 580
50 REM SCREEN*****
60 FOR A1=1 TO 5
70 FOR A2=1 TO 5
80 COLOR(A1,A2)=10*A1+90
90 CALL HCHAR(8+A1,11+A2,COLOR(A1,A2))
100 NEXT A2
110 NEXT A1
120 J$="FFFFFFFFFFFFFFFF"
130 FOR A=1 TO 5 :: CALL CHAR(10*A+90,J$):: NEXT A
140 CALL COLOR(9,16,1)
150 CALL COLOR(10,12,1)
160 CALL COLOR(12,10,1)
170 CALL COLOR(13,13,1)
180 CALL COLOR(14,5,1)
190 FOR T=1 TO 5
200 CALL HCHAR(8+T,11,ASC(STR$(T))): NEXT T
210 DISPLAY AT(14,10):"67890"

```



```
220 RETURN
230 REM RANDOMIZE*****
240 FOR T=1 TO 25
250 K=INT(RND*10+48)
260 GOSUB 300
270 NEXT T
280 RETURN
290 REM MOVE*****
300 ON (K-47)GOSUB 480,490,500,510,520,530,540,550,560,570
310 IF ROW<>0 THEN 400
320 FOR A=5 TO 1 STEP -1
330 COLOR(A+1,COLUMN)=COLOR(A,COLUMN)
340 NEXT A
350 COLOR(1,COLUMN)=COLOR(6,COLUMN)
360 FOR A=1 TO 5
370 CALL HCHAR(B+A,11+COLUMN,COLOR(A,COLUMN))
380 NEXT A
390 RETURN
400 FOR A=5 TO 1 STEP -1
410 COLOR(ROW,A+1)=COLOR(ROW,A)
```

```

420 NEXT A
430 COLOR(ROW,1)=COLOR(ROW,6)
440 FOR A=1 TO 5
450 CALL HCHAR(B+ROW,11+A,COLOR(ROW,A))
460 NEXT A
470 RETURN
480 ROW=0 :: COLUMN=5 :: RETURN
490 ROW=1 :: COLUMN=0 :: RETURN
500 ROW=2 :: COLUMN=0 :: RETURN
510 ROW=3 :: COLUMN=0 :: RETURN
520 ROW=4 :: COLUMN=0 :: RETURN
530 ROW=5 :: COLUMN=0 :: RETURN
540 ROW=0 :: COLUMN=1 :: RETURN
550 ROW=0 :: COLUMN=2 :: RETURN
560 ROW=0 :: COLUMN=3 :: RETURN
570 ROW=0 :: COLUMN=4 :: RETURN
580 REM MAIN PROGRAM*****
590 GOSUB 60
600 GOSUB 240
610 CALL KEY(S,K,S)
620 IF S=0 OR K<40 OR K>57 THEN 610
630 GOSUB 300
640 GOTO 610
650 END

```



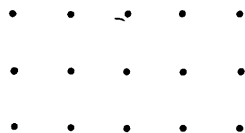
# Qui Vive

To be 'on the qui vive' means to be alert and watchful. This game is called QUI VIVE because to play it successfully you must always be on the look out for winning situations. The rules are very simple but the program needed to implement them is anything but. In fact, it presents quite a challenge to a games programmer.

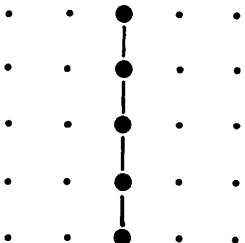
QUI VIVE was invented by Eugene de Wolf and is played on a square  $5 \times 5$  board. Each player has five checkers and must try to arrange them into one of seven symmetrical patterns. These are:



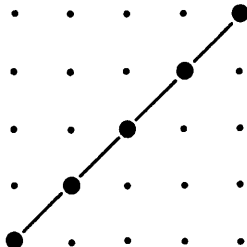
a horizontal line, e.g.



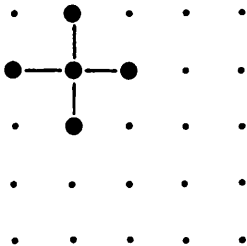
a vertical line, e.g.



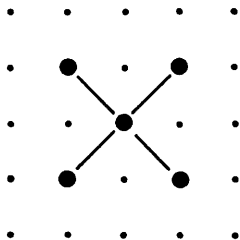
**a diagonal, e.g.**



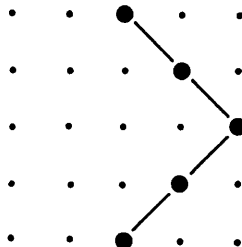
**a rectangular cross, e.g.**



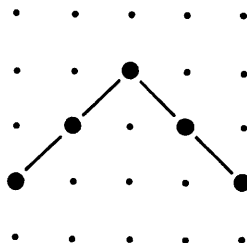
**a skew cross, e.g.**



a vertical wing, e.g.



a horizontal wing, e.g.



At the start of the game the board is empty and you and the computer take turns to place checkers on it. The positions on the board are labelled like this:

⑤	1	2	3	4	5
④	6	7	8	9	10
③	11	12	13	14	15
②	16	17	18	19	20
①	21	22	23	24	25
	Ⓐ	Ⓑ	Ⓒ	Ⓓ	Ⓔ

To place a checker at any particular position simply enter the appropriate coordinates: for instance for position 21 enter A1, for position 12, B3, etc. Your checkers are indicated by P and the computer's by C.

If all ten checkers have been placed on the board and no one has won, the game continues, with you and the computer taking turns to move checkers to try and gain a winning position. To move one of your checkers you enter the coordinates first of its present position and then of the position to which you wish to move it.

You must keep a look out for chances to make a winning pattern yourself, at the same time making sure that you are blocking any winning moves by the computer.

The program is one of the most interesting in this book. It contains a list of not only all 42 possible winning patterns but also over 100 particularly strong situations from which a player has a chance of making either one of two winning patterns. During the game the computer monitors this list in the light of the situation on the board and assesses what its best move is.

We advise you to play this game on an ordinary checkerboard against one of your friends before you take on the computer. You will soon see what a superb game it is.

```

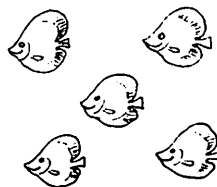
10 REM TI(EXTENDED)-QUI VIVE
20 CALL CLEAR
30 RANDOMIZE
40 DIM SF(41,4),SO(41)
50 DIM D(107,1),DF(15,8)
60 DIM CC(4,1),B(24),C(24)
70 GOTO 1870
80 REM compute points of df*****
90 GV=0 :: P1=0 :: P2=0 :: C1=0 :: C2=0
100 D1=B(DF(NF,0)):: D2=B(DF(NF,1))
110 IF D1=1 OR D2=1 THEN P1=1
120 IF D1=10 OR D2=10 THEN C1=10
130 D1=B(DF(NF,2)):: D2=B(DF(NF,3))
140 IF D1=1 OR D2=1 THEN P2=1
150 IF D1=10 OR D2=10 THEN C2=10
160 GV=B(DF(NF,4))+B(DF(NF,5))+B(DF(NF,6))
170 P=GV+P1+P2+C1+C2
180 IF GV=30 AND(C1=0 OR C2=0)THEN P=P-10
190 IF GV=3 AND(P1=0 OR P2=0)THEN P=P-1
200 RETURN
210 REM initialize single figures (sf)
220 FOR I=0 TO 41 :: READ X$
230 FOR J=1 TO 5
240 SF(I,J-1)=ASC(SEG$(X$,J,1))-65
250 NEXT J :: NEXT I :: RETURN
260 REM wings(AV<>)*****
270 DATA UQMSY,PLHNT,KGCIO
280 DATA AGMIE,FLRNJ,KQWSO
290 DATA AGMQU,BHNRV,CIOSW
300 DATA EIMSY,DHLRX,CGKQW
310 REM times(X)*****
320 DATA ACGKM,BDHLN,CEIMO
330 DATA FLHPR,GIMQS,HJNRT
340 DATA KMQUW,LNRVX,MOSWY
350 REM plus(+)******

```

```

360 DATA BFGHL,CGHIM,DHIJN
370 DATA GKLMQ,HLMN,IMNOS
380 DATA LPQRV,MQRSW,NRSTX
390 REM diagonals(\/)
400 DATA AGMSY,EIMQU
410 REM columns(I)*****
420 DATA AFKPU,BGLQV,CHMRW
430 DATA DINSX,EJOTY
440 REM rows(-)*****
450 DATA ABCDE,FGHIJ,KLMNO
460 DATA PQRST,UVWXY
470 REM initialize double figures (df)
480 FOR I=0 TO 107 STEP 12 :: READ X$
490 FOR J=0 TO 11 :: FOR K=0 TO 1
500 L=2*J+K+1 :: D(I+J,K)=ASC(SEG$(X$,L,1))-49
510 NEXT K :: NEXT J :: NEXT I
520 RETURN
530 DATA 171:1A1C1E1M1O1P2>202B2J
540 DATA 393<3=3?3G474:4=4?4A4G4O
550 DATA 4P505B5D5J696<6C6E6M7=7A
560 DATA 7C7I7O7P8>8B8D8J9?9E9K:?
570 DATA :A:E:K:O:P;>;0;D;J<=<C<I
580 DATA =G=I=O>F>H>J?G?K?P0F0J0L
590 DATA AGAIKAMAOAPBHBJBNCICMCP
600 DATA DJDLNEKEMEOFRFWGSGWHTHW
610 DATA IRIXJSJXKTKXLRLYMSMYNTNY
620 REM compute double figure fields*****
630 TC=4 :: TN=0
640 FOR I=0 TO 4 :: C(SF(D(DF1,0),I))=1 :: NEXT I

```



```

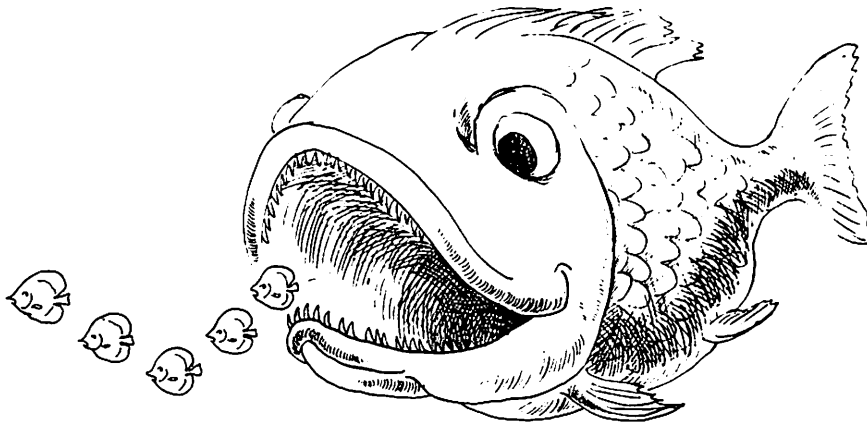
650 FOR I=0 TO 4 :: SF1=SF(D(DF1,1),I)
660 IF C(SF1)=1 THEN 680
670 DF(NF,TN)=SF1 :: TN=TN+1 :: GOTO 690
680 DF(NF,TC)=SF1 :: TC=TC+1 :: C(SF1)=0
690 NEXT I
700 FOR I=0 TO 4 :: SF1=SF(D(DF1,0),I)
710 IF C(SF1)=1 THEN DF(NF,TN)=SF1 :: TN=TN+1 :: C(SF1)=0
720 NEXT I :: RETURN
730 REM
740 REM display board*****
750 PRINT " "
760 FOR I=0 TO 4 :: FOR J=0 TO 4
770 F=5*I+J
780 IF B(F)=0 THEN PRINT " ";
790 IF B(F)=1 THEN PRINT "P";
800 IF B(F)=10 THEN PRINT "C";
810 NEXT J :: PRINT 5-I :: NEXT I
820 PRINT " "
830 PRINT "ABCDE"
840 PRINT " "
850 RETURN
860 REM determine owner of sf's*****
870 OM=-1 :: WM=-1
880 FOR SF1=0 TO 41
890 P=B(SF(SF1,0))+B(SF(SF1,1))+B(SF(SF1,2))+B(SF(SF1,3))+B(SF(SF1,4))
900 IF P=0 OR P=10 OR P=20 OR P=30 THEN SO(SF1)=10 :: GOTO 1000
910 IF P<>40 THEN 950
920 GOSUB 1010
930 GOSUB 1050
940 WM=1 :: SF1=41 :: GOTO 1000
950 IF P>20 OR P<3 OR P=11 OR P=12 THEN SO(SF1)=0 :: GOTO 1000
960 SO(SF1)=1
970 IF P=4 AND OM=-1 THEN GOSUB 1010
980 IF P=14 THEN GOSUB 1050
990 IF P=5 THEN SF1=41
1000 NEXT SF1 :: RETURN
1010 REM determine obligatory move*****
1020 FOR I=0 TO 4
1030 IF B(SF(SF1,I))=0 THEN OM=SF(SF1,I)
1040 NEXT I :: RETURN
1050 REM determine pinned checker*****

```

```

1060 FOR I=0 TO 4
1070 IF B(SF(SF1,I))<>10 THEN 1110
1080 FOR J=0 TO CC1-1
1090 IF SF(SF1,I)=CC(J,0)THEN CC(J,1)=1
1100 NEXT J
1110 NEXT I
1120 RETURN
1130 REM determine interesting df's*****
1140 NC=0 :: NP=0 :: NF=0
1150 FOR I=0 TO 15
1160 DF(I,8)=-1
1170 DF(I,7)=0
1180 NEXT I
1190 FOR DF1=0 TO 107
1200 O1=SO(D(DF1,0)):: O2=SO(D(DF1,1))
1210 IF O1<>O2 OR O1=0 OR O2=0 THEN 1330
1220 GOSUB 620
1230 GOSUB 80
1240 IF P=4 OR P=14 THEN DF(NF,7)=1 :: GOTO 1270
1250 IF O1=1 OR P<=DF(NF,8)THEN 1330
1260 DF(NF,7)=10

```





```

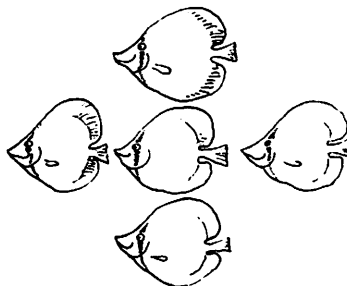
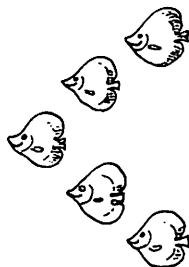
1270 DF(NF,8)=P
1280 NF=0
1290 FOR I=0 TO 15
1300 IF DF(I,8)>=DF(NF,8) THEN 1320
1310 IF DF(I,7)=0 OR DF(I,7)=10 THEN NF=I
1320 NEXT I
1330 NEXT DF1
1340 RETURN
1350 REM
1360 REM determine strategic value of move
1370 IF WM=1 THEN 1520
1380 MC=0 :: N2=0
1390 FOR NF=0 TO 15
1400 IF DF(NF,7)=0 THEN 1460
1410 GOSUB 80
1420 IF DF(NF,7)=1 THEN 1450
1430 IF P=MC THEN N2=N2+1
1440 IF P>MC THEN MC=P :: N2=1
1450 IF P=4 THEN NF=15
1460 NEXT NF
1470 IF PP=-1 THEN 1520
1480 IF P=4 THEN RETURN
1490 IF MC<MP THEN RETURN
1500 IF MC=MP AND N2<N1 THEN RETURN
1510 IF MC=MP AND N2=N1 AND RND<.5 THEN RETURN
1520 MP=MC :: N1=N2
1530 PP=CP :: PT=CT
1540 RETURN
1550 REM
1560 REM player move*****
1570 IF PC<5 THEN 1610
1580 PRINT "WHICH CHECKER DO YOU WANT TO MOVE"
1590 GOSUB 1690 :: PT=X
1600 IF B(PT)<>1 THEN PRINT "NOT POSSIBLE" :: GOTO 1580
1610 PRINT "WHERE DO YOU PUT YOUR CHECKER"
1620 GOSUB 1690 :: PP=X
1630 IF B(PP)<>0 THEN PRINT "NOT POSSIBLE" :: GOTO 1610
1640 IF PC=5 THEN B(PT)=0
1650 IF PC<5 THEN PC=PC+1
1660 B(PP)=1
1670 GOSUB 740

```

```

1680 RETURN
1690 REM input*****
1700 INPUT X$ :: IF LEN(X$)<>2 THEN 1750
1710 L$=SEG$(X$,1,1):: D$=SEG$(X$,2,1)
1720 IF L$<"A" OR L$>"E" OR D$<"1" OR D$>"5" THEN 1750
1730 X=ASC(L$)-5*VAL(D$)-40
1740 RETURN
1750 PRINT "WRONG INPUT,TRY AGAIN" :: GOTO 1700
1760 REM computer move*****
1770 IF CC1<5 THEN PT=CC1 :: CC1=CC1+1 :: GOTO 1830
1780 PRINT "I TAKE ";CHR$(65+PT-INT(PT/5)*5);5-INT(PT/5)
1790 B(PT)=0
1800 FOR CI=0 TO 4
1810 IF CC(CI,0)=PT THEN PT=CI :: CI=4
1820 NEXT CI
1830 PRINT "I PUT IT AT ";CHR$(65+PP-INT(PP/5)*5);5-INT(PP/5)
1840 B(PP)=10 :: CC(PT,0)=PP
1850 GOSUB 740
1860 RETURN
1870 REM main program*****
1880 PRINT "PLEASE WAIT"
1890 GOSUB 210
1900 GOSUB 470
1910 CC1=0 :: PC=0
1920 PP=12 :: GOTO 2140
1930 FOR I=0 TO 4 :: CC(I,1)=0 :: NEXT I

```



```

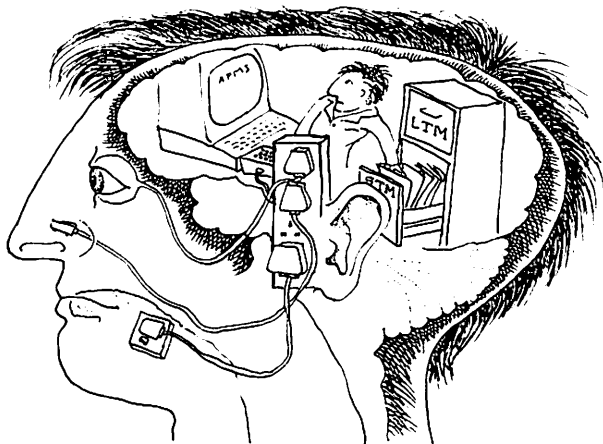
1940 GOSUB 860
1950 IF P=5 THEN END
1960 IF WM=1 THEN 1960
1970 GOSUB 1130
1980 MP=0 :: N1=0 :: PP=-1
1990 BF=0 :: EF=24
2000 IF OM<>-1 THEN BF=OM :: EF=OM
2010 FOR CP=BF TO EF
2020 IF B(CP)<>0 THEN 2130
2040 B(CP)=10
2050 IF CC1<5 THEN GOSUB 1360 :: GOTO 2120
2060 FOR CI=0 TO 4
2070 IF CC(CI,1) THEN 2110
2080 CT=CC(CI,0):: B(CT)=0
2090 GOSUB 1360
2100 B(CT)=10
2110 NEXT CI
2120 B(CP)=0
2130 NEXT CP
2140 GOSUB 1760
2150 IF WM=1 THEN END
2160 GOSUB 1560
2170 PRINT "PLEASE WAIT"
2180 GOTO 1930

```

# STM

The nature of memory is certainly one of the most interesting topics in psychology. To remember something has been defined as 'to show in present responses some signs of earlier learned responses.' Why, though, do we remember some things and forget others? We may recognize someone we haven't seen in years, or call to mind a tune having heard only a few notes of it, but find it very difficult to remember, for instance, a telephone number. There are in fact two distinct types of memory: short term memory (STM) and long term memory (LTM). If you want to find out more about this intriguing topic we recommend that you read *Introduction to Psychology* by Hilgard & Atkinson.

This game tests your short term memory. Play it with your friends and family and see who can remember most. You will see this 'menu' on the screen:



- DO YOU WANT TO PLAY WITH
1. LETTERS?
  2. NUMBERS?
  3. 0 OR 1 ONLY?
  4. OR DO YOU WANT TO STOP?

Now you make your choice by entering 1, 2, 3 or 4, and also choose a level of difficulty. Say you entered 1 to play with letters. A letter will now appear on the screen but only for a very short time. You have to enter that letter. The computer will then show you two letters which you have to enter, then three, and so on. Obviously, as the number of letters increases, remembering them all becomes more difficult. What is the longest string of characters you can remember? Can you beat our record of eight?

```
10 REM TI-STM
20 CALL CLEAR
30 RANDOMIZE
40 B=0
50 R$=""
60 PRINT "Do you want to play with"
70 PRINT "1:letters?"
80 PRINT "2:numbers?"
90 PRINT "3:0 and 1 only?"
100 PRINT "4:or do you want to stop"
110 CALL KEY(5,K,S)
120 IF S=0 THEN 110
130 IF K>57 THEN 110
140 IF K<48 THEN 110
150 K=K-48
160 ON K GOSUB 420,450,480,510
170 INPUT "Difficulty?(1-6)":DIF
180 REM DISPLAY*****
190 B=B+1
200 R$=""
210 CALL CLEAR
220 FOR CC1=1 TO B
```

```

230 A=INT(RND*NR)+W
240 CALL HCHAR(12,16,A)
250 R$=R$&CHR$(A)
260 FOR CC2=1 TO 100/DIF
270 NEXT CC2
280 CALL HCHAR(12,16,31)
290 FOR CC2=1 TO 100/DIF
300 NEXT CC2
310 NEXT CC1
320 REM INPUT ANSWER*****
330 INPUT "ANSWER: ":AN$
340 IF AN$=R$ THEN 190
350 PRINT "WRONG"
360 PRINT "YOUR SCORE IS: ";B-1
370 PRINT "THE ANSWER IS: ";R$
380 PRINT "PRESS ANY KEY"
390 CALL KEY(5,K,S)
400 IF S=0 THEN 390
410 GOTO 20
420 NR=26
430 W=65
440 RETURN
450 NR=10
460 W=48
470 RETURN
480 NR=2
490 W=48
500 RETURN
510 PRINT "End of this game"
520 END

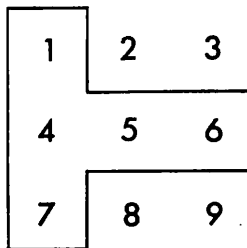
```

# One to Five

This exasperating game, which we saw first on a CASIO calculator, takes quite some logical reasoning to solve. It is played on a board with nine positions set out like this:

1	2	3
4	5	6
7	8	9

A horizontal and a vertical line drawn through any one position will cross five positions in all. For instance lines drawn through position 4 will also cross positions 1, 5, 6, and 7.



The positions on the board are occupied by a random pattern of digits between 0 and 5. If you enter one of the positions (using the key shown on the screen) the digit on that position, and all the digits on the horizontal and vertical lines going through that position, are increased by 1 (except 5 which becomes 0). For instance if we had

2	2	4
---	---	---

1	5	2
---	---	---

1	5	4
---	---	---

and entered 4, the board would change to

3	2	4
---	---	---

2	0	3
---	---	---

2	5	4
---	---	---

The puzzle is solved when the board looks like this:

0	0	0
---	---	---

0	0	0
---	---	---

0	0	0
---	---	---



```

10 REM TI-ONE TO FIVE
20 RANDOMIZE
30 REM DISPLAY*****
40 CALL CLEAR
50 IL$="234WERASD"
60 FOR T3=0 TO 2
70 FOR T4=0 TO 2
80 CALL HCHAR(T3+10,T4+13,ASC(STR$(A(T3,T4))))
90 CALL HCHAR(T3+10,T4+19,ASC(SEG$(IL$,3*T3+T4+1,1)))
100 NEXT T4
110 NEXT T3
120 FOR T1=0 TO 10
130 I=INT(RND*9)+1
140 GOSUB 250
150 NEXT T1
160 M=0
170 REM INPUT*****
180 CALL KEY(1,I,STA)
190 IF (I<1)+(I>9)<>0 THEN 180
200 IF STA=0 THEN 180
210 M=M+1
220 GOSUB 250
230 GOTO 180
240 REM EXECUTION MOVE*****
250 X=INT((9-I)/3+1)
260 Y=((I-1)/3-INT((I-1)/3))*3+1
270 FOR T2=0 TO 2
280 A(X-1,T2)=A(X-1,T2)+1
290 A(T2,Y-1)=A(T2,Y-1)+1
300 NEXT T2
310 A(X-1,Y-1)=A(X-1,Y-1)-1
320 T=0
330 FOR T3=0 TO 2
340 FOR T4=0 TO 2
350 IF A(T3,T4)<>6 THEN 370
360 A(T3,T4)=0
370 IF A(T3,T4)<>0 THEN 390
380 T=T+1
390 CALL HCHAR(T3+10,T4+13,ASC(STR$(A(T3,T4))))
400 NEXT T4
410 NEXT T3

```

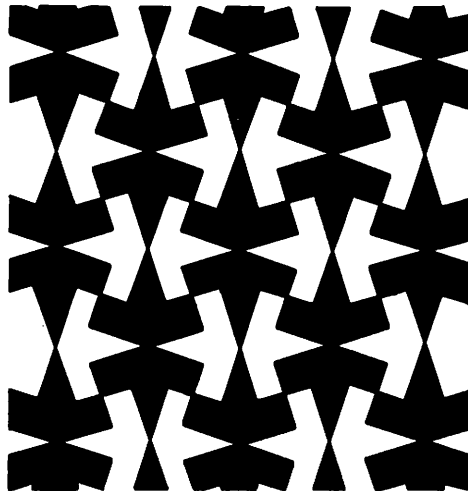
```
420 IF T<9 THEN 430 ELSE 450
430 RETURN
440 REM PROBLEM SOLVED*****
450 PRINT "THAT'S GREAT"
460 PRINT "FOUND IN";M;"MOVES"
470 END
```

# Escher

The work of the Dutch graphic artist M.C. Escher (1902-1972), based as it is on symmetry and mathematical forms, appeals particularly to computer programmers.

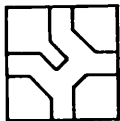
Now with the help of the computer you can produce your own patterns based on the same principles used by Escher.

Consider the following pattern:

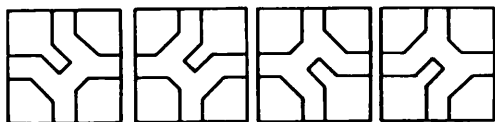


It is based on a mosaic in the Alhambra palace in Spain. If you look closely you can see that all the tiles are the same shape and that they are arranged so that the 'inverse video' of the white tiles shows the same pattern rotated

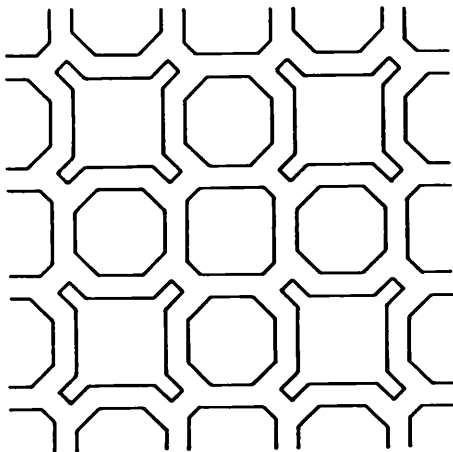
through 90 degrees. Escher studied such patterns and invented a game based on them. He devised a patterned tile such as



which he rotated through four positions



and arranged to form a continuous pattern.



This program does something very similar. After you have filled a 'basic' square with symbols the computer rotates and shifts it to fill the screen with your own 'Escher'.

At the start of the game the computer displays the shapes that can be used to compose your basic square. Then the computer requests

SIZE OF BASIC SQUARE (3 — 8)

Now if you enter 3, for instance, you have to fill nine positions with shapes. When you have done this the computer asks

DIRECTION OF SQUARE (0 — 3):?

Each number corresponds to one of the basic directions. This question will be repeated many times, as you will see, after which the computer will display your undoubtedly fantastic creation.

```

10 REM TI-ESCHER GAME
20 DATA 0000000000000000,0000000000000000,0000000000000000,0000000000000000
30 DATA 8080808080808080,FF00000000000000,0101010101010101,00000000000000FF
40 DATA 2020202020202020,0000FF0000000000,0404040404040404,0000000000FF0000
50 DATA 1010101010101010,000000FF00000000,8080808080808080,00000000FF000000
60 DATA C0C0C0C0C0C0C0C0,FFFF000000000000,0303030303030303,000000000000FFFF
70 DATA F0F0F0F0F0F0F0F0,FFFFFFFF00000000,0F0F0F0F0F0F0F0F,00000000FFFFFFFF
80 DATA 8040201008040201,0102040810204080,8040201008040201,0102040810204080
90 DATA 101020C000000000,8080403000000000,0000000030408080,00000000C0201010
100 DATA FFFEFCF8F0E0C080,FF7F3F1F0F070301,0103070F1F3F7FFF,80C0E0F0F8FCFEFF
110 DATA F0F0F0F000000000,0F0F0F0F00000000,000000000F0F0F0F,00000000F0F0F0F0
120 DATA 00183C7E7E3C1800,00183C7E7E3C1800,00183C7E7E3C1800,00183C7E7E3C1800
130 DATA 8142241818244281,8142241818244281,8142241818244281,8142241818244281
140 DATA 08080808FF080808,10101010FF101010,101010FF10101010,080808FF08080808
150 DATA AA55AA55AA55AA55,55AA55AA55AA55AA,AA55AA55AA55AA55,55AA55AA55AA55AA
160 DATA F0F0F0F0F0F0F0F0,F0F0F0F0FF0F0F0F0,F0F0F0F0F0F0F0F0,F0F0F0F0FF0F0F0F0
170 FOR TEL=96 TO 155
180 READ X*
```

```

190 CALL CHAR(TEL,X$)
200 NEXT TEL
210 REM INPUT BASIC SQUARE*
220 PRINT "YOU HAVE THE FOLLOWING CHARACTERS:"
230 FOR TEL=96 TO 155 STEP 4
240 PRINT TEL;
250 CALL HCHAR(24,10,TEL)
260 PRINT
270 NEXT TEL
280 INPUT "SIZE OF BASIC SQUARE(3-8)":GR
290 IF (GR<3)+(GR>8)<>0 THEN 280
300 HH=INT(24/GR)
310 HHH=HH*HH
320 DIM D(64)
330 DIM RI(64)
340 II=0
350 FOR CC=1 TO GR*GR
360 PRINT "GRAPHIC NR:";CC;
370 INPUT D(CC)
380 IF (D(CC)<96)+(D(CC)>155)+(D(CC)/4-INT(D(CC))/4<>0)<>0 THEN 360
390 NEXT CC
400 REM INPUT DIRECTIONS***
410 FOR CC=1 TO HHH
420 PRINT "DIRECTION OF SQUARE ";CC;"(0-3): ";
430 INPUT RI(CC)
440 IF (RI(CC)<0)+(RI(CC)>3)<>0 THEN 420
450 NEXT CC
460 REM DRAW ESCHER PICTURE*
470 CALL CLEAR
480 AAA=0
490 FOR OB1=1 TO HH*GR STEP GR
500 FOR OB2=4 TO HH*GR STEP GR
510 AAA=AAA+1
520 I=0
530 ON RI(AAA)+1 GOSUB 580,640,700,760
540 NEXT OB2
550 NEXT OB1
560 GOTO 560
570 END
580 FOR C1=OB1 TO OB1+GR-1
590 FOR C2=OB2 TO OB2+GR-1

```

```

600 GOSUB 820
610 NEXT C2
620 NEXT C1
630 RETURN
640 FOR C2=OB2+GR-1 TO OB2 STEP -1
650 FOR C1=OB1 TO OB1+GR-1
660 GOSUB 820
670 NEXT C1
680 NEXT C2
690 RETURN
700 FOR C1=OB1+GR-1 TO OB1 STEP -1
710 FOR C2=OB2+GR-1 TO OB2 STEP -1
720 GOSUB 820
730 NEXT C2
740 NEXT C1
750 RETURN
760 FOR C2=OB2 TO OB2+GR-1
770 FOR C1=OB1+GR-1 TO OB1 STEP -1
780 GOSUB 820
790 NEXT C1
800 NEXT C2
810 RETURN
820 I=I+1
830 CALL HCHAR(C1,C2,D(I)+RI(AAA))
840 RETURN

```

# Genius at Work

Play this simple game to find out how good you are at thinking mathematically. The computer displays six numbers and a larger 'target' number. You must pick two of the numbers and one of the four simple mathematical functions:

addition +  
subtraction —  
multiplication ×  
division ÷

The numbers you have chosen will be manipulated in this way. For instance, if you entered 7, 18 and +, the computer calculates  $7 + 18 = 25$ . The numbers 7 and 18 are then replaced in the initial list by 25. You now choose again, and can carry on making such choices until a fixed time limit expires. When this happens the computer will add up the numbers in the list and display the difference between this total and the target number. Your aim is to make this difference as small as possible.

The character of this game is simplicity, so the whole family can play. Now you can find out if there is a genius in your family!

```
10 REM TI-GENIUS AT WORK
20 RANDOMIZE
30 FOR B=1 TO 4
40 NR(B)=INT(RND*9+1)
50 NEXT B
60 NR(5)=INT(RND*5+1)*10
70 NR(6)=INT(RND*4+1)*25
80 TARGET=INT(RND*600+200)
90 TI=26
100 REM MAIN PROGRAM*****
```





```

110 CALL CLEAR
120 PRINT "THE TARGET NUMBER IS:";TARGET
130 REM EXECUTE*****
140 FOR B=1 TO 6
150 PRINT NR(B);
160 NEXT B
170 PRINT
180 GOSUB 360
190 REM**EXECUTE
200 TH=0
210 AWAY=TARGET
220 FOR B=1 TO 6
230 AWAY=AWAY-NR(B)
240 WH=WH+(TARGET-NR(B)=0)
250 TH=TH+(NR(B)=0)
260 NEXT B
270 IF WH<>-1 THEN 310
280 PRINT "Congratulations!You got it."
290 END
300 IF TH<=-5 THEN 320
310 IF TI>0 THEN 110
320 PRINT "You failed"
330 PRINT "The difference is:";ABS(AWAY)
340 END
350 REM INPUT*****
360 PRINT "Select a number: ";
370 GOSUB 790
380 IF TI<1 THEN RETURN
390 NR1=0
400 FOR A=1 TO 6
410 IF NR(A)<>IN THEN 430
420 NR1=A
430 NEXT A
440 IF NR1<>0 THEN 470
450 PRINT "You don't have this number"
460 GOTO 360
470 PRINT "Select another number: ";
480 GOSUB 790
490 IF TI<1 THEN RETURN
500 NR2=0
510 FOR A=1 TO 6

```

```

520 IF NR(A)<>IN THEN 550
530 IF A=NR1 THEN 550
540 NR2=A
550 NEXT A
560 IF NR2<>0 THEN 590
570 PRINT "You don't have this number"
580 GOTO 470
590 PRINT "Select an operation: ";
600 GOSUB 790
610 AW$=CHR$(K)
620 PRINT AW$
630 IF AW$<>"+" THEN 660
640 NR(NR1)=NR(NR1)+NR(NR2)
650 GOTO 770
660 IF AW$<>"*" THEN 690
670 NR(NR1)=NR(NR1)*NR(NR2)
680 GOTO 770
690 IF AW$<>"/" THEN 720
700 NR(NR1)=NR(NR1)/NR(NR2)
710 GOTO 770
720 IF AW$<>"-" THEN 750
730 NR(NR1)=NR(NR1)-NR(NR2)
740 GOTO 770
750 PRINT "Wrong entry operation"
760 GOTO 590
770 NR(NR2)=0
780 RETURN
790 REM*INPUT*
800 IN=0
810 RET=0
820 IF D=100 THEN 840
830 IF D<>0 THEN 900
840 TI=TI-1
850 IF TI<0 THEN RETURN
860 CALL HCHAR(12,31,TI+97)
870 CALL SOUND(10,880,0)
880 D=0
890 D=D+1
900 CALL KEY(3,K,S)
910 IF S<>0 THEN GOSUB 960
920 IF RET=1 THEN CALL HCHAR(11,31,31)

```

```
930 IF RET=1 THEN RETURN
940 IF D<>100 THEN GOTO 890
950 GOTO 840
960 REM*CALCULATE INPUT*
970 IF K<48 THEN 1020
980 IF K>57 THEN 1020
990 PRINT USING "##":(K-48);
1000 IN=10*IN+K-48
1010 GOTO 1030
1020 RET=1
1030 IF K=13 THEN PRINT
1040 RETURN
```

# Shark Hunt

For five days now you have been adrift in your boat searching the seven seas. Suddenly you see a slight ripple on the mirror-like surface of the ocean — there it is, your quarry, that terror of the deep — the shark. Use the following keys to move your boat

W	E	R
S		D
Z	X	C

For instance, pressing E moves your boat northwards, that is, up the screen. Your echo-sounder will tell you how close you are to the shark — the nearer you get the higher the note it gives. Stray too far from the shark and you are told

**SORRY; YOU WENT TOO FAR**

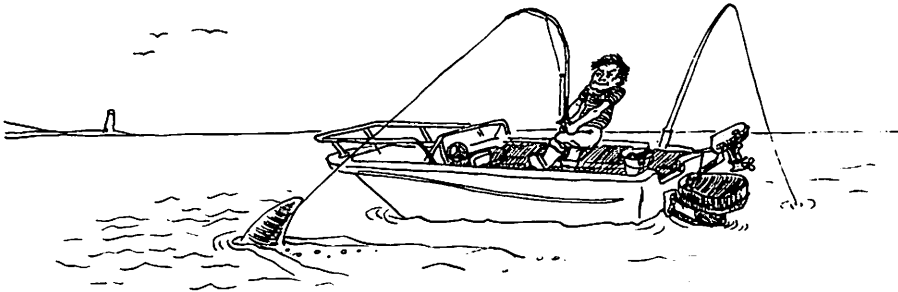
When you think you are close enough you can take a shot at the shark by pressing the space bar. If you hit the target you will see the message

**THAT'S IT! CONGRATULATIONS!**

Unfortunately you only have 200 ergs of energy to use. Moving your boat and firing your gun both use energy. If you use up all your energy the computer tells you

## YOU RAN OUT OF ENERGY

At the end of the game you will be given information on your performance as a shark hunter.

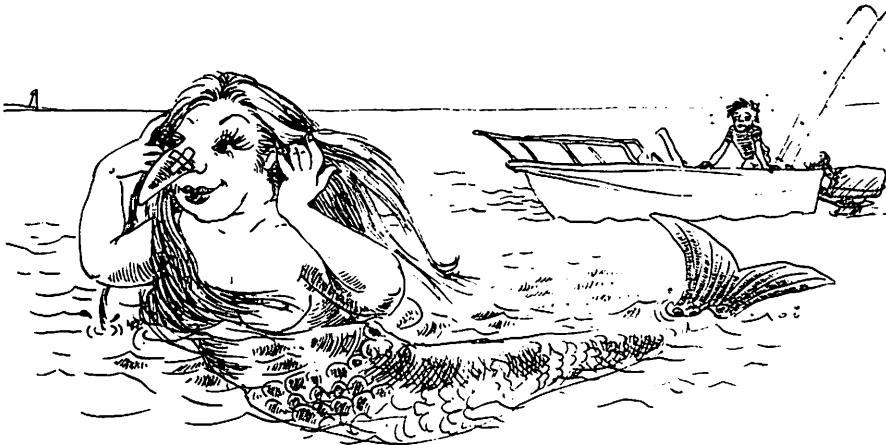


```
10 REM TI-SHARK HUNT
20 DIM C0$(3)
30 NG=0
40 TR=0
50 MR=0
60 FOR COM=1 TO 3
70 READ C0$(COM)
80 NEXT COM
90 DATA SORRY YOU WENT TOO FAR
100 DATA THAT'S IT CONGRATULATIONS!
110 DATA YOU RAN OUT OF ENERGY
120 GOTO 830
130 REM INITIALIZE*****
140 CALL CLEAR
150 PRINT "          WER"
160 PRINT "          S D"
170 PRINT "          ZXC"
180 FOR I=1 TO 8
190 PRINT
```

```

200 NEXT I
210 NG=NG+1
220 SX=INT(RND*70-35)
230 SY=INT(RND*70-35)
240 DI=ABS(SX)+ABS(SY)
250 SD=DI
260 MD=DI
270 DATA 200,0,0,0,0
280 READ EN,HI,NS,VX,VY
290 RETURN
300 REM INPUT SPEED*****
310 CALL KEY(3,K,S)
320 VX=3*((K=87)+(K=83)+(K=90)-(K=82)-(K=68)-(K=67))
330 VY=3*((K=90)+(K=88)+(K=67)-(K=87)-(K=69)-(K=82))
340 SH=(K=32)
350 RETURN
360 REM COMPUTE*****
370 SX=SX-VX
380 SY=SY-VY
390 SX=SX+SGN(SX)
400 SY=SY+SGN(SY)
410 EN=EN+(VX<>0)+(VY<>0)-1
420 IF SH=0 THEN 460
430 NS=NS+1
440 HI=(RND*DI<2)
450 EN=EN-2*DI
460 EN=-(EN>0)*EN
470 IF HI THEN 500
480 DI=ABS(SX)+ABS(SY)
490 MD=-(MD<DI)*MD-(MD>DI)*DI
500 RETURN
510 REM OUTPUT*****
520 IF SH=0 THEN 550
530 CALL SOUND(100,110,0)
540 GOTO 560
550 CALL SOUND(100,3000-25*DI,0)
560 X$=STR$(EN)
570 D=LEN(X$)
580 FOR T=1 TO 4
590 IF T<=D THEN 600 ELSE 620
600 CALL HCHAR(2,14+T,ASC(SEG$(X$,T,1)))

```



```
610 GOTO 630
620 CALL HCHAR(2,14+T,32)
630 NEXT T
640 RETURN
650 REM  END OF GAME*****
660 PRINT C0$(CA)
670 PRINT "STARTING DISTANCE:";SD;"M"
680 PRINT "NUMBER OF SHOTS:";NS
690 PRINT "YOU APPROACHED WITHIN ";MD;"M"
700 RE=50-DI
710 IF HI=0 THEN 740
720 PRINT "YOU HIT FROM:";DI;"M"
730 RE=50+EN/4+SD/10
740 TR=TR+RE
750 IF RE<MR THEN 770
760 MR=RE
770 PRINT "REWARD ON A SCALE OF 100:"
780 PRINT "THIS GAME:";INT(RE)
790 PRINT "AVERAGE OVER";NG;"GAMES:";INT(TR/NG)
800 PRINT "MAXIMUM:";INT(MR)
```



```
810 RETURN
820 REM  MAIN PROGRAM*****
830 GOSUB 130
840 GOSUB 300
850 GOSUB 360
860 GOSUB 510
870 CA=-3*(EN<=0)-2*HI-(DI>100)
880 CA=CA+3*(CA>3)
890 IF CA=0 THEN 840
900 GOSUB 650
910 INPUT "ANOTHER GAME?(Y/N)":AN$
920 IF SEG$(AN$,1,1)="Y" THEN 940
930 END
940 RESTORE 270
950 GOTO 830
```

# Shakespearian Shuffle

Shakespeare, one of the world's greatest writers, and chess, the king of games, are combined in this unusual puzzle.

Letters are arranged on a chessboard like this:

T	O		B	E		O	R
N	O	T		T	O		B
E		T	H	A	T		I
S		T	H	E		Q	U
E	S	T	I	O	N		
W	I	L	L	I	A	M	
S	H	A	K	E	S	P	E
A	R	E		I	6	O	3


A chesspiece, the knight, then moves around the board as it would in a normal game of chess. As it jumps from one square to another, the letters or symbols on the squares are exchanged. In this way the text on the board is jumbled up. The computer will ask you

LEVEL?

and you enter the number of moves you want the knight to make. Obviously,

the higher this number the more mixed up the board becomes.

You will be shown the board after the knight has made his moves. Your task is to unscramble the text by moving the knight back around the board. Enter a number between 1 and 8 to move the knight, as shown.

	1		2	
8				3
				
7				4
	6		5	

```

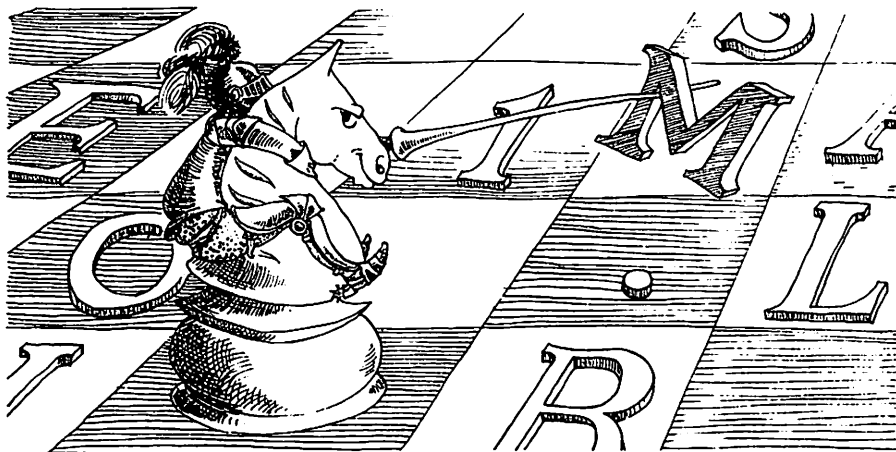
10 REM TI-SHAKESPEARIAN SHUFFLE
20 DIM HX(8),HY(8),B$(64),T$(8)
30 RANDOMIZE
40 GOTO 560
50 REM INITIALIZE*****
60 XO=1
70 YO=1
80 FOR K=1 TO 8
90 READ T$(K)
100 FOR J=1 TO 8
110 B$((K-1)*8+J)=SEG$(T$(K),J,1)
120 NEXT J
130 NEXT K
140 FOR K=1 TO 8
150 READ HX(K),HY(K)
160 NEXT K
170 RETURN
180 REM KNIGHT'S MOVE*****
190 DONE=0
200 IF (XS+XO)>8 THEN 350

```

```

210 IF XS+X0<1 THEN 350
220 IF YS+Y0>8 THEN 350
230 IF YS+Y0<1 THEN 350
240 DONE=1
250 XN=X0+XS
260 YN=Y0+YS
270 REM CHANGE LETTERS*****
280 P=XN+(YN-1)*8
290 V=X0+(Y0-1)*8
300 W$=B$(P)
310 B$(P)=B$(V)
320 B$(V)=W$
330 X0=XN
340 Y0=YN
350 RETURN
360 REM DISPLAY*****
370 CALL CLEAR
380 PRINT
390 PRINT "12345678"

```



```

400 FOR K=1 TO 8
410 FOR J=1 TO 8
420 PRINT B*((K-1)*8+J);
430 NEXT J
440 PRINT K
450 NEXT K
460 PRINT
470 PRINT " 1 2 "
480 PRINT "8   3"
490 PRINT "    "
500 PRINT "7   4"
510 PRINT " 6 5 "
520 PRINT
530 PRINT
540 RETURN
550 REM MAIN PROGRAM*****
560 GOSUB 50
570 INPUT "LEVEL=":L
580 FOR K=1 TO L
590 M=INT(RND*8+1)
600 XS=HX(M)
610 YS=HY(M)
620 GOSUB 180
630 IF DONE=0 THEN 590
640 NEXT K
650 GOSUB 360
660 PRINT "X-OLD=";X0
670 PRINT "Y-OLD=";Y0
680 GOSUB 760
690 GOSUB 180
700 IF DONE=0 THEN 680
710 GOTO 650
720 DATA "TO BE OR","NOT TO B",E THAT I","S THE QU
730 DATA "ESTION. ","WILLIAM ","SHAKESPE","ARE 1603"
740 DATA 2,-1,2,1,-1,2,1,2,-2,1,-2,-1,-1,-2,1,-2
750 END
760 REM INPUT*****
770 INPUT Q
780 IF (Q<1)+(Q>8)<>0 THEN 770
790 XS=(Q=1)+(Q=6)+2*(Q=8)+2*(Q=7)-(Q=2)-(Q=5)-2*(Q=3)-2*(Q=4)

```

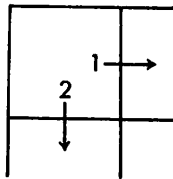
```
800 YS=(Q=8)+(Q=3)+2*(Q=1)+2*(Q=2)-(Q=7)-(Q=4)-2*(Q=6)-2*(Q=5)
810 RETURN
```

# Explosion

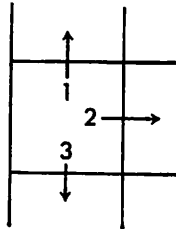
Most computer versions of EXPLOSION set out the board for two or more opponents to play on. With this program you are pitted against the computer itself. Are you up to the challenge?

EXPLOSION is played on a board of  $3 \times 3$  or  $4 \times 4$  squares. The computer will ask you to enter the size of the board you want.

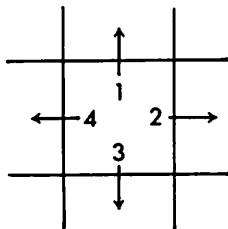
Each square on the board has a capacity equal to the number of squares directly adjoining it. This means that corner squares have a capacity of 2



edge squares have a capacity of 3



and central squares have a capacity of 4.



You and the computer have checkers, of opposite colors, that you place on the board in turn. You may place a checker on any empty square, or on any square which already has one or more of your own checkers on it. A square will 'explode' when the number of checkers it carries reaches its capacity. An exploding square empties, its checkers spreading out, one to each neighboring square. These checkers will 'take over' any opposing checkers on these squares.

As the game progresses the explosions get bigger and bigger. Eventually the whole board will explode in one color. If it's your color you've won!

```

10 REM TI(EXTENDED)-EXPLOSION
20 RANDOMIZE
30 DIM RB(5,5),SB(5,5),ST(5,5)
40 GOTO 1620
50 FOR X=1 TO SI
60 FOR Y=1 TO SI
70 SB(X,Y)=RB(X,Y)
80 NEXT Y
90 NEXT X
100 RETURN
110 FOR X=1 TO SI
120 FOR Y=1 TO SI
130 RB(X,Y)=SB(X,Y)
140 NEXT Y
150 NEXT X

```



```

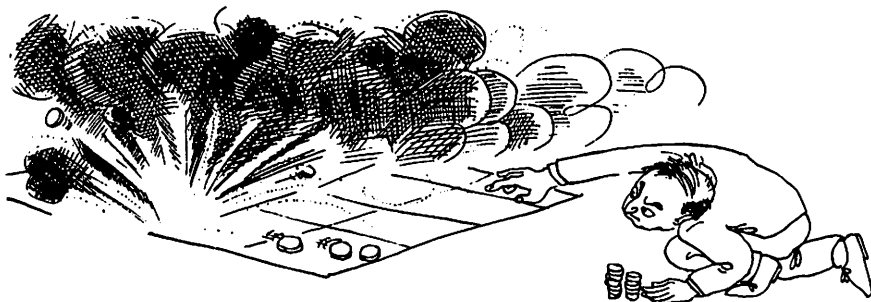
160 RETURN
170 REM **INITIALIZATION**
180 INPUT "ENTER SIZE OF BOARD (3 OR 4)":SI
190 IF SI<3 THEN 180
200 IF SI>4 THEN 180
210 FOR X=1 TO SI
220 FOR Y=1 TO SI
230 ST(X,Y)=4+(X=1)+(X=SI)+(Y=1)+(Y=SI)
240 NEXT Y
250 NEXT X
260 INPUT "WHO STARTS? YOU OR ME:":AN$
270 CM=(SEG$(AN$,1,1)="Y")
280 CALL CLEAR
290 FOR I=1 TO SI
300 CALL HCHAR(3,3*I+11,I+48)
310 CALL HCHAR(3,3*I+10,88)
320 NEXT I
330 FOR Y=1 TO SI
340 CALL HCHAR(2*Y+4,8,89)
350 CALL HCHAR(2*Y+4,9,Y+48)
360 FOR X=1 TO SI
370 A=X
380 B=Y
390 GOSUB 1710
400 NEXT X
410 NEXT Y
420 DISPLAY AT(17,1):"X="
430 DISPLAY AT(18,1):"Y="
440 RETURN
450 REM **INPUT AND EXECUTE MOVE**
460 CALL HCHAR(17,5,32)
470 CALL HCHAR(18,5,32)
480 CALL SOUND(50,1000,0)
490 CALL KEY(3,MX,S)
500 IF S=0 THEN 490
510 CALL HCHAR(17,5,MX)
520 MX=MX-48
530 IF MX<1 OR MX>SI THEN 490
540 FOR I=0 TO 40
550 NEXT I
560 CALL KEY(3,MY,S)

```

```

570 IF S=0 THEN 560
580 CALL HCHAR(18,5,MY)
590 MY=MY-48
600 IF MY<1 OR MY>SI THEN 490
610 IF RB(MX,MY)<0 THEN 490
620 GOSUB 50
630 X=MX
640 Y=MY
650 DI=-1
660 GOSUB 700
670 IF ET THEN 1570
680 GOSUB 110
690 RETURN
700 REM **OUTPUT EXPLOSION**
710 SB(X,Y)=SB(X,Y)+PL
720 NE=0
730 A=X
740 B=Y
750 IF DI THEN GOSUB 1710
760 XP=0
770 FOR X=1 TO SI
780 FOR Y=1 TO SI
790 IF ABS(SB(X,Y))<ST(X,Y) THEN 980
800 XP=-1
810 NE=NE+1
820 SB(X,Y)=SB(X,Y)-ST(X,Y)*PL
830 A=X
840 B=Y
850 IF DI THEN GOSUB 1710
860 EX=X
870 EY=Y+1
880 GOSUB 1040
890 EX=X+1
900 EY=Y
910 GOSUB 1040
920 EX=X
930 EY=Y-1
940 GOSUB 1040
950 EX=X-1
960 EY=Y
970 GOSUB 1040

```



```

980 NEXT Y
990 NEXT X
1000 ET=(NE>SI*SI)
1010 IF ET THEN 1030
1020 IF XP THEN 760
1030 RETURN
1040 REM **ADD TO SURROUNDING**
1050 SB(EX,EY)=PL*(ABS(SB(EX,EY))+1)
1060 A=EX
1070 B=EY
1080 IF NOT DI THEN 1100
1090 IF ST(EX,EY)>0 THEN GOSUB 1710
1100 RETURN
1110 REM **COMPUTER MOVES**
1120 BE=1000
1130 FOR TX=1 TO SI
1140 FOR TY=1 TO SI
1150 IF RB(TX,TY)>0 THEN 1320
1160 GOSUB 50
1170 X=TX
1180 Y=TY
1190 DI=0
1200 GOSUB 700
1210 IF NOT ET THEN 1250
1220 MX=TX
1230 MY=TY
1240 GOTO 1340

```

```

1250 GOSUB 1430
1260 IF EN>BE THEN 1320
1270 IF EN<BE THEN 1290
1280 IF RND>.5 THEN 1320
1290 BE=EN
1300 MX=TX
1310 MY=TY
1320 NEXT TY
1330 NEXT TX
1340 REM **ACTUAL MOVE**
1350 GOSUB 50
1360 X=MX
1370 Y=MY
1380 DI=-1
1390 GOSUB 700
1400 IF ET THEN 1570
1410 GOSUB 110
1420 RETURN
1430 REM **EVALUATE SITUATION**
1440 EN=0
1450 FOR X=1 TO SI
1460 FOR Y=1 TO SI
1470 EN=EN+SB(X,Y)
1480 IF -SB(X,Y)<ST(X,Y)-1 THEN 1540
1490 EN=EN-2
1500 IF SB(X+1,Y)=ST(X+1,Y)-1 THEN EN=EN+10
1510 IF SB(X,Y+1)=ST(X,Y+1)-1 THEN EN=EN+10
1520 IF SB(X-1,Y)=ST(X-1,Y)-1 THEN EN=EN+10
1530 IF SB(X,Y-1)=ST(X,Y-1)-1 THEN EN=EN+10
1540 NEXT Y
1550 NEXT X
1560 RETURN
1570 REM **END**
1580 PRINT "AN ETERNAL EXPLOSION          RESULTED, SO ";
1590 IF PL=1 THEN PRINT "YOU";ELSE PRINT "I";
1600 PRINT " HAVE WON IN";NT;"TURNS"
1610 END
1620 REM **MAIN PROGRAM**
1630 GOSUB 170
1640 IF CM THEN 1670
1650 PL=1

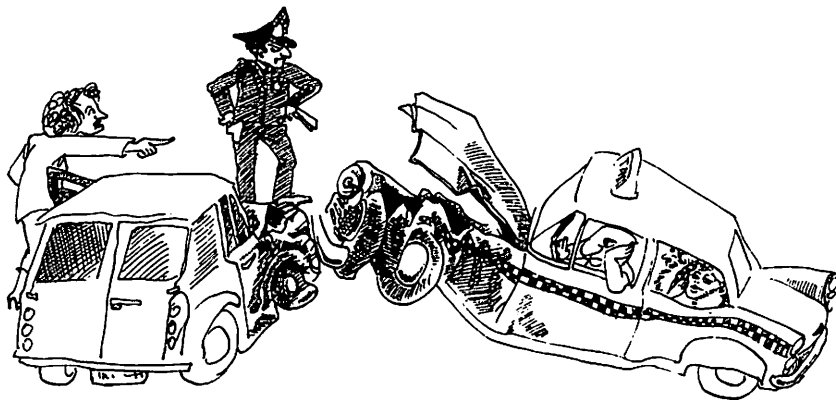
```

```
1660 GOSUB 450
1670 PL=-1
1680 GOSUB 1110
1690 NT=NT+1
1700 GOTO 1650
1710 REM **PRINT SB(A,B)**
1720 IF SB(A,B)<0 THEN C=45 ELSE C=32
1730 CALL HCHAR(2*B+4,3*A+10,C)
1740 CALL HCHAR(2*B+4,3*A+11,ASC(STR$(ABS(SB(A,B)))))
1750 RETURN
```

# New York, New York

This original and exciting game puts you in a helicopter high over New York, looking down on the city's streets. At the moment they are deserted, but soon traffic will appear, and it's up to you to keep it moving. You do this by controlling the city's traffic lights. When a car (represented by a square) reaches a red light it will stop. Each traffic light bears a symbol — pressing the key bearing that symbol changes the light to green and the car will continue on its journey.

At the beginning of the game you have 100 points. You gain 10 points every time a car goes through a crossroads, and 25 points for every car that gets right across the city. However if two cars collide at a crossroads you lose 100 points and the road is temporarily blocked. If a car has to stop because of a collision or a red light you also lose points. If you run out of points the game ends — if not it lasts a fixed time. The more points you get, the better traffic cop you are!



```

10 REM TI(EXTENDED)
20 REM NEW YORK - NEW YORK
30 REM MY RECORD IS 2270 POINTS
40 REM
50 AW=40 :: DIM CX(40),CY(40),CR(40),PX(37),PY(37),P$(37),AX(3),AY(3),FX(9),FY(9)
)
60 RESTORE :: CALL CLEAR :: CALL CHARSET :: CALL SCREEN(15):: PU=100
70 FOR I=0 TO 40 :: CR(I)=0 :: NEXT I
80 TIME=0 :: TL=0 :: AF=0
90 FOR I=32 TO 44 :: READ X$ :: CALL CHAR(I,X$):: NEXT I
100 DATA "55AA55AA55AA55AA","252A25EAE52A252A","55AA5500FF181818","54A454A757A45
4A4","181818FF00AA55AA"
110 DATA "1818181818181818","000000FFFF000000","181818FFFF181818","FFFFFFFFFFFFFF
FFF"
120 DATA "0F0F0F0F0F0F0F0F","F0F0F0F0F0F0F0F0","00000000FFFFFFFF","FFFFFFFF000000
000"
130 FOR I=48 TO 95 :: CALL CHARPAT(I,X$):: CALL CHAR(48+I,X$):: NEXT I
140 CALL COLOR(1,16,2)! ROADS
150 CALL COLOR(2,6,2)! CARS
160 FOR I=3 TO 8 :: CALL COLOR(I,2,7):: CALL COLOR(6+I,2,4):: NEXT I ! TRAFFIC L
IGHTS
170 CALL HCHAR(5,1,38,32):: CALL HCHAR(13,5,38,24):: CALL HCHAR(20,1,38,32)
180 CALL VCHAR(1,5,37,24):: CALL VCHAR(5,13,37,8):: CALL VCHAR(5,20,37,16):: CAL
L VCHAR(1,28,37,24)
190 FOR I=1 TO 17 :: READ X,Y,CH :: CALL HCHAR(Y,X,CH):: NEXT I
200 DATA 13,4,34,20,4,34,5,5,39,13,5,39,20,5,39,28,5,39
210 DATA 4,13,35,5,13,39,13,13,39,20,13,39,28,13,39,29,13,33,13,14,36
220 DATA 5,20,39,20,20,39,28,20,39,20,21,36
230 FOR I=0 TO 37 :: READ PX(I),PY(I),P$(I):: CALL HCHAR(PY(I),PX(I),ASC(P$(I)))
:: NEXT I
240 DATA 4,3,A,27,3,B,7,4,C,15,4,D,22,4,E,30,4,F,3,6,G,11,6,H,18,6,I,26,6,J
250 DATA 6,7,K,14,7,L,21,7,M,29,7,N,4,11,0,12,11,P,19,11,Q,27,11,R
260 DATA 7,12,S,15,12,T,22,12,U,11,14,V,18,14,W,26,14,X,6,15,Y,21,15,Z,29,15,0
270 DATA 4,18,1,19,18,2,27,18,3,7,19,4,22,19,5,30,19,6,3,21,7,18,21,8,26,21,9,6,
22,<,29,22,>
280 X$="NEW_YORK" :: DISPLAY AT(2,6):X$;
290 FOR I=1 TO 8 :: CALL HCHAR(2,17+I,ASC(SEG$(X$,I,1))+48):: NEXT I
300 DISPLAY AT(16,7):"YOU HAVE";:: DISPLAY AT(17,6):"100 POINTS";
310 RANDOMIZE
320 DEF FNR(X)=INT(RND*X)
330 AX(0)=1 :: AX(1)=-1 :: AY(2)=1 :: AY(3)=-1

```





```

340 FOR I=0 TO AW :: R=CR(I):: GOSUB 950 :: IF R=0 THEN 440
350 X=CX(I):: Y=CY(I)
360 IF R AND 1 THEN X=X+1 :: F=43 :: 00=44 :: 0=38 :: SX=0 :: SY=1 :: IF X>32 TH
EN 480
370 IF R AND 2 THEN X=X-1 :: F=44 :: 00=43 :: 0=38 :: SX=0 :: SY=-1 :: IF X<1 TH
EN 480
380 IF R AND 4 THEN Y=Y+1 :: F=42 :: 00=41 :: 0=37 :: SX=-1 :: SY=0 :: IF Y>24 T
HEN 480
390 IF R AND 8 THEN Y=Y-1 :: F=41 :: 00=42 :: 0=37 :: SX=1 :: SY=0 :: IF Y<1 THE
N 480
400 IF R AND 16 THEN 450
410 IF R AND 32 THEN 560
420 IF R AND 64 THEN 0=39 :: R=R-64 :: CR(I)=R :: PU=PU+10
430 GOTO 490
440 NEXT I :: GOTO 660
450 CALL GCHAR(CY(I)+SY,CX(I)+SX,LIGHT)
460 IF LIGHT>95 THEN CR(I)=(R OR 32)AND NOT 16 :: GOTO 490
470 PU=PU-1 :: GOTO 440
480 PU=PU+25 :: PN=PN+1 :: CR(I)=0 :: GOTO 550
490 CALL GCHAR(Y,X,PS):: IF PS=F OR PS=40 THEN PU=PU-1 :: GOTO 440
500 CALL GCHAR(Y+SY,X+SX,A):: IF A<>32 THEN CR(I)=R OR 16
510 IF PS=00 THEN F=40
520 CALL GCHAR(CY(I),CX(I),V):: IF V=40 THEN 0=00
530 IF V=32 THEN CR(I)=0 :: A0=A0+1 :: PU=PU-100 :: GOTO 440
540 CALL HCHAR(Y,X,F)
550 CALL HCHAR(CY(I),CX(I),0):: CX(I)=X :: CY(I)=Y :: GOTO 440
560 CALL GCHAR(Y,X,A):: IF A=F THEN 440
570 IF A<>39 THEN 630
580 VB=0 :: FOR J=0 TO 3 :: A=2^J :: CALL GCHAR(Y+AY(J),X+AX(J),B):: IF NOT(B=38
OR B=37)THEN VB=VB OR A
590 NEXT J
600 B=2^FNR(4):: IF (B AND VB)=0 THEN CR(I)=B OR 64 :: GOTO 540
610 IF VB=15 THEN 440
620 GOTO 600
630 A0=A0+1 :: CR(I)=0 :: IF A<>32 THEN FX(AF)=X :: FY(AF)=Y :: AF=AF+1 :: PU=PU
-100
640 F=32 :: CALL SOUND(4000,-6,0)
650 GOTO 540
660 IF CR(TL)=0 THEN 690
670 TL=TL+1 :: IF TL>AW THEN TL=0

```

```

680 GOTO 780
690 A=FNR(4):: CR(TL)=2^A :: ON A+1 GOTO 700,710,720,730
700 CX(TL)=1 :: GOTO 740
710 CX(TL)=32 :: GOTO 740
720 CY(TL)=1 :: GOTO 760
730 CY(TL)=24 :: GOTO 760
740 IF FNR(2)=1 THEN CY(TL)=5 :: GOTO 780
750 CY(TL)=20 :: GOTO 780
760 IF FNR(2)=1 THEN CX(TL)=5 :: GOTO 780
770 CX(TL)=28 :: GOTO 780
780 REM TIME CONTROL
790 IF PU<0 THEN PU=0
800 DISPLAY AT(17,5):USING "####":PU;
810 IF PU=0 THEN 880
820 TIME=TIME+1 :: IF TIME>100 THEN 880
830 IF FNR(25)>0 OR AF=0 THEN 870
840 CALL HCHAR(FY(0),FX(0),39)
850 FOR I=0 TO 8 :: FY(I)=FY(I+1):: FX(I)=FX(I+1):: NEXT I
860 AF=AF-1
870 GOTO 340
880 IF PU=0 THEN CALL SOUND(4000,-1,0):: GOTO 900
890 FOR I=1 TO 5 :: CALL SOUND(1000,698,0):: CALL SOUND(1000,784,0):: CALL SOUND
(1000,880,0):: NEXT I
900 DISPLAY AT(2,6):"PLAY_AGAIN?_Y|N Y " :: ACCEPT AT(2,23)SIZE(-1)VALIDATE("YN
"):A$
910 IF A$="N" THEN 930
920 GOTO 60
930 CALL CLEAR :: CALL CHARSET
940 PRINT "YOU HAVE";PU;"POINTS" :: END
950 REM SUBROUTINE KEY PRESSED ?
960 CALL KEY(0,LIGHT,CH)
970 IF CH<>1 THEN RETURN
980 IF LIGHT>64 AND LIGHT<91 THEN NR=LIGHT-65 :: GOTO 1030
990 IF LIGHT>47 AND LIGHT<58 THEN NR=LIGHT-22 :: GOTO 1030
1000 IF LIGHT=44 THEN NR=36 :: GOTO 1030
1010 IF LIGHT=46 THEN NR=37 :: GOTO 1030
1020 RETURN
1030 CALL GCHAR(PY(NR),PX(NR),COLLIGHT)
1040 IF COLLIGHT>95 THEN NEWCH=COLLIGHT-48 :: GOTO 1060
1050 NEWCH=COLLIGHT+48

```

1060 CALL HCHAR(PY(NR),PX(NR),NEWCH)  
1070 CALL SOUND(-200,1047,0)  
1080 RETURN

# Key

Searching for a key you have lost can be an aggravating experience at the best of times, but when you have to find it as quickly as possible and, what's more, it's hidden inside a computer the whole thing becomes very exasperating, but also very challenging.

The computer has stored inside it a string of 20 ones and zeroes.

1 0 1 0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 1 0

These figures can be shifted cyclically, that is, digits are moved from the righthand end to the left. This is done three times and the numbers in each column added, for instance

row 1	→	0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 1 0 1 0 1
row 2	→	0 1 1 1 0 1 0 1 0 1 1 0 1 0 1 0 0 1 0 1
row 3	→	0 1 0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 1 0 1
sums	→	0 2 2 1 2 1 2 2 2 3 0 3 0 3 1 1 3 0 3

This is all kept hidden from you: it is in fact the key you must find.

What you are shown are the three rows of numbers, each shifted again. For instance if the top row is shifted 18 positions, the middle row 16 positions and the bottom row 12 positions you will see

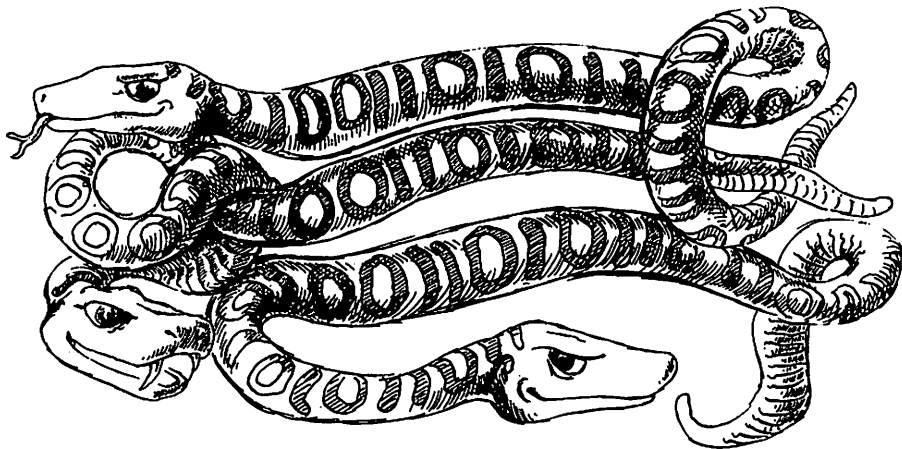
1 0 1 0 1 1 1 0 1 0 1 0 1 1 0 1 0 1 0 0
0 1 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1
1 1 1 0 1 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0
2 0 0 0 0 1 1 2 1 1 1 2 3 3 1 0 1 2 2

You are not shown the sum of the new columns but the difference for each column, between this sum and the first 'hidden' sum. In the first column, for instance, the hidden sum was 0 and the new sum is 2 so you see 2; in the second column the hidden sum was 2 and so is the new sum so you see 0. You must now shift the three rows until they are the same as the hidden key, when, of course, all the differences will be 0. The rows are numbered (the top row is 1, the middle 2 and the bottom 3) so that if you enter

ROW = 2  
STEPS = -1

the middle row will shift one position to the right.

How few turns will it take you to find the key? It has been done in as few as 10.



```

10 REM TI-KEY
20 DIM T(4,20)
30 A$="10100101011101010110"
40 GOTO 520
50 REM RANDOMIZE*****
60 RANDOMIZE
70 FOR J=1 TO 3
80 R=INT(RND*19+1)
90 FOR K=1 TO 20
100 DIS=K+R
110 IF DIS<21 THEN 130
120 DIS=DIS-20
130 T(J,K)=VAL(SEG$(A$,DIS,1))
140 NEXT K
150 NEXT J
160 RETURN
170 REM COMPUTE KEY*****
180 FOR K=1 TO 20
190 T(4,K)=T(1,K)+T(2,K)+T(3,K)
200 NEXT K
210 RETURN
220 REM SCREEN*****
230 CALL CLEAR
240 TEL=0
250 FOR K=1 TO 20
260 CALL HCHAR(10,K+2,ASC(STR$(T(1,K))))
270 CALL HCHAR(11,K+2,ASC(STR$(T(2,K))))
280 CALL HCHAR(12,K+2,ASC(STR$(T(3,K))))
290 V=T(4,K)-T(3,K)-T(2,K)-T(1,K)
300 V=ABS(V)
310 IF V<>0 THEN 330
320 TEL=TEL+1
330 CALL HCHAR(14,K+2,ASC(STR$(V)))
340 NEXT K
350 RETURN
360 REM INPUT*****
370 INPUT "ROW(1-3):":J
380 IF (J<1)+(J>3)>0 THEN 370
390 INPUT "STEPS(1-19):":S
400 IF (S<1)+(S>19)>0 THEN 390
410 NT=NT+1

```

```

420 RETURN
430 REM SHIFT ROW*****
440 FOR I=1 TO S
450 H=T(J,20)
460 FOR K=19 TO 1 STEP -1
470 T(J,K+1)=T(J,K)
480 NEXT K
490 T(J,1)=H
500 NEXT I
510 RETURN
520 REM MAIN PROGRAM*****
530 GOSUB 50
540 GOSUB 170
550 GOSUB 50
560 GOSUB 220
570 IF TEL=20 THEN 610
580 GOSUB 360
590 GOSUB 430
600 GOTO 560
610 PRINT "FOUND IN ";NT;"TURNS"
620 END

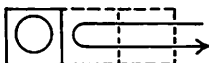
```

# Black Box

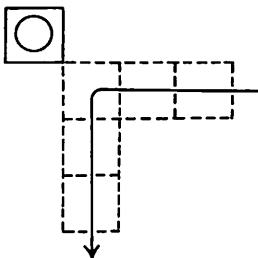
Armed only with a laser beam you must discover the whereabouts of a handful of atoms hidden in a vast black box.

The box consists of  $8 \times 8 \times 8$  cubes. Atoms can be hidden in any of the cubes, apart from those in the outer layer. However, there are never more than five atoms in the box. These atoms will reflect or divert laser beams according to the following rules:

— a beam which strikes an atom is reflected straight back in the opposite direction



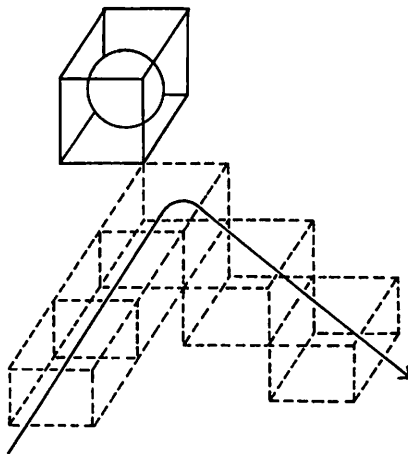
— a beam which is set to pass through a cube directly adjacent to an atom will be reflected at right angles



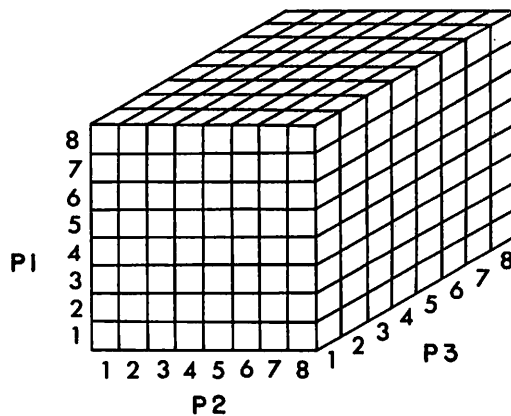
— a beam which is set to pass through a cube diagonally adjacent to an atom will be reflected in a direction which can be obtained by adding two



right-angled reflections



The box is numbered like this:



The computer will ask you to enter P1, P2, and P3, the coordinates of the position where you want the beam to enter the box. Obviously, this must be on the surface, so at least one of the coordinates must be 1 or 8. When you have entered the shot the computer will tell you where the beam has emerged. Remember that a beam may be reflected by more than one atom. After you have had a few shots you should be able to work out where the atoms are — but then how few is ‘a few’?

```

10 REM TI-BLACK BOX
20 RANDOMIZE
30 FOR T=1 TO 5
40 M(INT(RND*6+2),INT(RND*6+2),INT(RND*6+2))=1
50 NEXT T
60 PRINT "PLEASE INPUT SHOT"
70 INPUT "P1=":P1
80 INPUT "P2=":P2
90 INPUT "P3=":P3
100 S1=(P1=8)-(P1=1)
110 S2=(P2=8)-(P2=1)
120 S3=(P3=8)-(P3=1)
130 IF (P1=8)+(P1=1)+(P2=8)+(P2=1)+(P3=8)+(P3=1)=-1 THEN 160
140 PRINT "WRONG INPUT"
150 GOTO 60
160 FOR A1=-1 TO 1
170 FOR A2=-1 TO 1
180 FOR A3=-1 TO 1
190 IF M(P1+A1,P2+A2,P3+A3)<>1 THEN 230
200 S1=S1-A1
210 S2=S2-A2
220 S3=S3-A3
230 NEXT A3
240 NEXT A2
250 NEXT A1
260 S1=SGN(S1)
270 S2=SGN(S2)
280 S3=SGN(S3)
290 P1=P1+S1
300 P2=P2+S2

```

```

310 P3=P3+S3
320 IF (P1=0)+(P1=9)+(P2=0)+(P2=9)+(P3=0)+(P3=9)=0 THEN 160
330 PRINT "RESULT:";P1-S1;P2-S2;P3-S3
340 INPUT "SHOOT OR GUESS? (S OR G) ":AW$
350 IF AW$="S" THEN 60
360 IF AW$="G" THEN 380
370 END
380 PRINT "ENTER COORDINATES"
390 INPUT "P1=":P1
400 INPUT "P2=":P2
410 INPUT "P3=":P3
420 IF M(P1,P2,P3)=1 THEN 450
430 PRINT "WRONG"
440 GOTO 340
450 PRINT "RIGHT"
460 GOTO 340

```

# Treasure Hunt

Have you ever dreamed of going in search of hidden treasure? Of journeying through wild and hostile countryside, living off the land and sleeping out, until you reach the remote and forbidding land where your glittering prize is hidden? If so, this is the game for you.

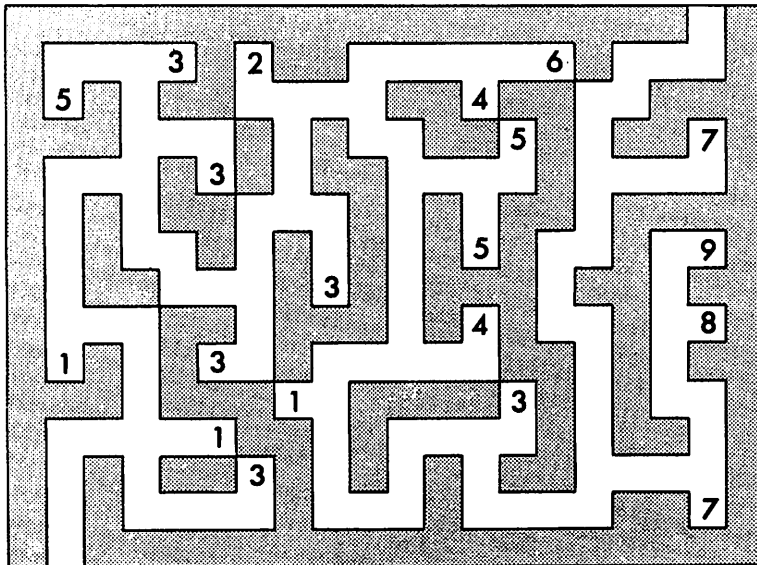
You will be taken high up into the Rocky Mountains and given a map which shows where the treasures you seek are hidden, and what they are worth. To reach them you must make your way along narrow twisting paths — one



false step means certain death — by using the cursor keys. You discover the treasures simply by treading on them.

But wait . . . it's not that easy. You didn't really think it would be that simple, did you? The treasures you seek are very carefully guarded by some extremely vicious and terrifying birds who will attack you if you are not careful. If they get you in their sights you will be paralyzed with fear. The only thing you can do to try and stop them is to press the space bar.

If you still feel up to the challenge start your search in the lower left hand corner of the map. Collect as much treasure as you can and take it away through the upper righthand exit. This will give you bonus points, and another chance to journey along the paths, picking up treasure. In fact you can make as





```

340 NEXT C2
350 NEXT C1
360 RETURN
370 REM PLAYER MOVES*****
380 CALL KEY(3,K,S)
390 X1=X+(K=69)-(K=88)
400 Y1=Y+(K=83)-(K=68)
410 IF (B(X1,Y1)=120)+(X1<1)+(X1>15)+(Y1<1)+(Y1>20)<>0 THEN 460
420 CALL HCHAR(X+5,Y+5,B(X,Y))
430 X=X1
440 Y=Y1
450 CALL HCHAR(X+5,Y+5,136)
460 RETURN
470 REM BIRD ATTACKS*****
480 IF RND>Q THEN 610
490 FOR CC=25 TO 6 STEP -1
500 CALL HCHAR(X+5,CC,B(X,CC-5))
510 CALL KEY(3,K,S)
520 IF K=32 THEN 550
530 IF CC<>Y+5 THEN 560
540 BS=-1
550 CC=1
560 NEXT CC
570 FOR CC=6 TO 25
580 CALL HCHAR(X+5,CC,B(X,CC-5))
590 NEXT CC
600 CALL HCHAR(X+5,Y+5,136)
610 RETURN
620 REM INITIALIZE*****
630 NG=NG+1
640 X=15
650 Y=2
660 Q=Q+.1
670 RETURN
680 REM MAIN PROGRAM*****
690 GOSUB 40
700 GOSUB 620
710 GOSUB 370
720 IF (X=1)+(Y=19)=-2 THEN 690
730 IF B(X,Y)=31 THEN 760
740 SCORE=SCORE+NG*(B(X,Y)-48)

```

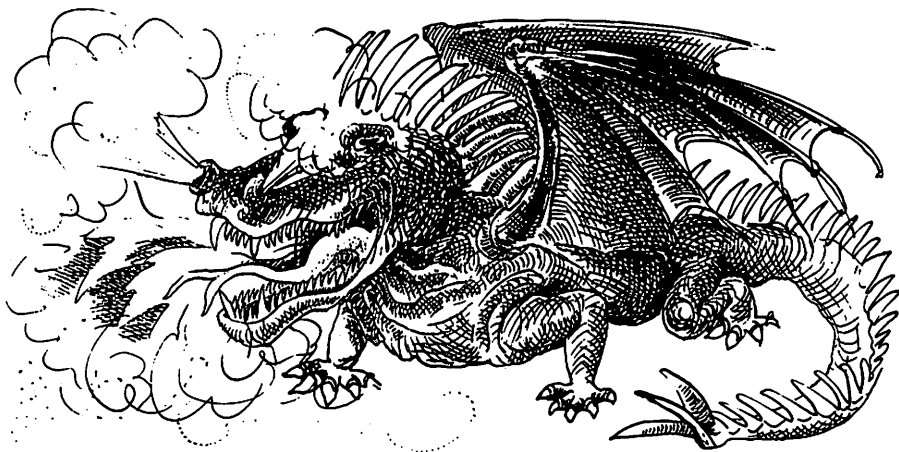
```
750 B(X,Y)=31
760 GOSUB 470
770 IF BS=0 THEN 710
780 PRINT "YOU FOUND";SCORE;"DOLLARS"
790 END
```



# I.T. — The Adventure of the Century

In this crazy adventure you will sink into the bowels of the earth and meet a very strange creature who lives there. His name is, of course, I.T. which stands for Intra Terrestrial.

If you have never played an adventure game before don't worry about the rules — there aren't any! You have to work everything out for yourself as you journey beneath the earth's surface, performing extraordinary tasks and facing terrifying dangers as you go. If it all gets too much for you enter HELP and see what the computer comes up with.



It will be quite some time after you have entered the RUN command before anything appears on the screen. It is important that the ALPHA LOCK key is not depressed during the game, because all your entries must be in lower case. In almost all situations, entering the first character of a word is sufficient, for instance for north you can just enter n.

So, type in the game, take out a good insurance policy, give the RUN command, and off you go!

```
10 REM TI(EXTENDED)-I.T.
20 RANDOMIZE
30 DIM K(36,6),R$(36),H$(15),VP(12),V$(12),B(12),PR(6,3)
40 GOSUB 2810
50 IHD=8 :: NKAM=0
60 GOTO 80
70 GOSUB 130
80 GOSUB 390
90 GOSUB 290
100 IF CWD<5 THEN 70
110 PRINT "You were killed"
120 END
130 NKAM=0
140 IHD=0
150 INPUT "What would you like to do:":C1$
160 LNG=LEN(C1$)
170 FOR I=1 TO 15
180 IF SEG$(H$(I),1,LNG)=C1$ THEN IHD=I :: I=16
190 NEXT I
200 IF IHD>6 THEN RETURN
210 IF IHD=0 THEN 260
220 NKAM=K(IKM,IHD)
230 IF NKAM<>0 THEN RETURN
240 PRINT "Impossible direction"
250 GOTO 130
260 PRINT "I do not understand "
270 GOTO 130
280 RETURN
290 PRINT "you are in the ";R$(IKM)
300 IF IKM>24 OR BLL=1 THEN 330
```

```

310 PRINT "You cannot see much here"
320 GOTO 340
330 IF IVW<>0 THEN PRINT "There is a ";V$(IVW);" here"
340 FOR I=1 TO 6
350 IF K(IKM,I)<>0 THEN PRINT "You can go ";H$(I)
360 NEXT I
370 PRINT
380 RETURN
390 IF NKAM<>0 THEN IKM=NKAM
400 IVW=0
410 FOR I=1 TO 12
420 IF VP(I)=IKM THEN IVW=I :: I=12
430 NEXT I
440 IPS=0
450 FOR I=1 TO 6
460 IF PR(I,1)=IKM THEN IPS=I :: I=6
470 NEXT I
480 IF IKM<12 AND CRI<>8 THEN GOSUB 2670
490 CLL=CLL+BLL
500 IF IHD<7 THEN 530
510 ON IHD-6 GOSUB 560,760,810,910,1130,1220,1340,1420,1500
520 REM
530 IF IPS=0 THEN RETURN
540 ON IPS GOSUB 1620,1840,2080,2180,2290,2470
550 RETURN
560 IF IPS=0 OR IPS>4 THEN PRINT "Killing time is the only      thing you can do h
ere" :: RETURN
570 IVWB=7
580 GOSUB 2610
590 IF IBZ=0 THEN PRINT "You can't attack without      sword" :: RETURN
600 PRINT "So you want to kill the ";P$(IPS)
610 PRINT "You attack it with your      sword";
620 IF RND<.5 THEN PRINT " but you missed him" :: GOTO 650
630 PRINT " and you give your victim a terrible blow"
640 PR(IPS,3)=PR(IPS,3)-1
650 IF RND>.2 THEN 700
660 PRINT "You can hit it again but be quick"
670 INPUT "Do you want to?(y or n)";Y$
680 IF Y$<>"y" THEN 700
690 PRINT "You raise your sword again";:: GOTO 620
700 IF PR(IPS,3)>1 THEN RETURN

```

```

710 IF PR(IPS,3)=1 THEN PRINT "Your victim is badly wounded" :: RETURN
720 PRINT "You killed your victim"
730 PR(IPS,1)=0 :: IVW=8+IPS
740 VP(IVW)=IKM :: IPS=0
750 RETURN
760 PRINT "Possible commands:"
770 FOR I=1 TO 15
780 PRINT H$(I)
790 NEXT I
800 RETURN
810 IF IVW<>0 THEN 840
820 PRINT "There is nothing to take    here"
830 RETURN
840 IF BLL=0 AND IKM<25 THEN PRINT "You cannot take things that you do not see"
:: RETURN
850 PRINT "I take the ";V$(IVW);" for you"
860 B(IBMAX)=IVW
870 IBMAX=IBMAX+1
880 VP(IVW)=0
890 IVW=0
900 RETURN
910 INPUT "What would you like to drop? ":D$
920 LNG=LEN(D$)
930 IVWB=0
940 FOR I=1 TO 12
950 IF D$=SEG$(V$(I),1,LNG) THEN IVWB=I :: I=12
960 NEXT I
970 IF IVWB<>0 THEN 1000
980 PRINT "I do not understand you"
990 RETURN
1000 GOSUB 2610
1010 IF IBZ<>0 THEN 1040
1020 PRINT "Impossible to drop"
1030 RETURN
1040 PRINT "I dropped the ";V$(B(IBZ));" for you"
1050 IVW=B(IBZ)
1060 VP(B(IBZ))=IKM
1070 B(IBZ)=B(IBMAX-1)
1080 IBMAX=IBMAX-1
1090 IF IKM<>25 THEN RETURN
1100 INPUT "Do you want to drop more?(y or n)":Y$

```

```

1110 IF Y$="y" THEN 910
1120 RETURN
1130 IF IBMAX<>1 THEN 1160
1140 PRINT "You own nothing"
1150 GOTO 1200
1160 PRINT "You have the following      things:"
1170 FOR I=1 TO IBMAX-1
1180 PRINT V$(B(I))
1190 NEXT I
1200 PRINT "You can still have ";5-CWD;"wounds before you die"
1210 RETURN
1220 IVWB=6
1230 GOSUB 2610
1240 IF IBZ<>0 THEN 1270
1250 PRINT "Impossible without bandage"
1260 RETURN
1270 PRINT "I use all the bandages to      bandage you"
1280 CALL SOUND(100,1000,0)
1290 PRINT "So, that will hold for a      while"
1300 CWD=0
1310 B(IBZ)=B(IBMAX-1)
1320 IBMAX=IBMAX-1
1330 RETURN
1340 IVWB=1
1350 GOSUB 2610
1360 IF IBZ<>0 THEN 1390
1370 PRINT "Impossible without a lantern"
1380 RETURN
1390 PRINT "The lantern is on now"
1400 BLL=1
1410 RETURN
1420 IVWB=1
1430 GOSUB 2610
1440 IF IBZ<>0 THEN 1470
1450 PRINT "You do not even have a      lantern"
1460 RETURN
1470 PRINT "The lantern is off now"
1480 BLL=0
1490 RETURN
1500 IVWB=4
1510 GOSUB 2610

```



```

1520 IF IBZ<>0 THEN 1550
1530 PRINT "Nothing to read"
1540 RETURN
1550 IF BLL=1 OR IKM>24 THEN 1580
1560 PRINT "It is to dark to read here"
1570 RETURN
1580 T$="Uifsf!jt!b!sfdjqu!gps!!!!!!dppljft!jo!uijt!cppl" :: GOSUB 3520
1590 T$="Ju!tbzt!ublf!b!efbe!!!!!!ifmmipvoe!boe!tpnf!xifbu/!!!Espq!uijt!po!b!
cbscfdvf!boe!xbju!b!njovuf" :: GOSUB 3520
1600 PRINT "That is all"
1610 RETURN
1620 IF IVW=3 THEN PR(1,2)=3 :: VP(IVW)=0 :: IVW=0
1630 ON PR(1,2)GOTO 1640,1690,1720,1790,1820
1640 PRINT "I.T.(the intra-terrestrial) is here"

```

```

1650 PRINT "It looks like he wants to talk"
1660 PRINT "He talks but you cannot understand him"
1670 PR(1,2)=2
1680 RETURN
1690 T$="J/U/!nblft!hftuvsft!uibu!if!offet!tpnfuijoh!up!esjol" :: GOSUB 3520
1700 T$="If!mpeplt!sbuifs!eftqfsbuf" :: GOSUB 3520
1710 RETURN
1720 T$="J/U/!offefe!uif!esjol!wfsz!!nvdi/" :: GOSUB 3520
1730 T$="If!tbzt!QMFBTF!hp!epxo!joup!uif!dbuft!boe!gJoe!uif!!!!!!ovmmjuz!cpnc" :: GOSUB 3520
1740 T$="Ju!nbz!cz!op!xbz!fyqmpef/!B!npotufs!qspufdut/" :: GOSUB 3520
1750 T$="Pomz!ZPV!dbo!tbwf!uif!fbsui/" :: GOSUB 3520
1760 PRINT "Suddenly I.T. collapses and falls down"
1770 PR(1,2)=4
1780 RETURN
1790 PRINT "I.T. is in a coma."
1800 PR(1,2)=5
1810 RETURN
1820 PRINT "I.T. is here": "He is in a coma"
1830 RETURN
1840 IF IVW=8 THEN PR(2,2)=5 :: VP(IVW)=0 :: IVW=0
1850 ON PR(2,2)GOTO 1860,1900,1930,1960,2000,2060
1860 PRINT "There is an enormous monsterhere."
1870 PRINT "The monster yells:are you a cookie"
1880 PR(2,2)=2
1890 RETURN
1900 PRINT "The monster yells louder andlouder:ARE YOU A COOKIE"
1910 PR(2,2)=3
1920 RETURN
1930 PRINT "The monster keeps yelling and becomes rather aggressive"
1940 PR(2,2)=4
1950 RETURN
1960 PRINT "The monster gives you a terrible blow. Your head is spinning"
1970 CWD=CWD+1
1980 PR(2,2)=INT(RND*3)+2
1990 RETURN
2000 PRINT "The monster starts to eat atonce."
2010 PRINT "It falls asleep"
2020 CALL SOUND(100,110,0)
2030 K(16,6)=1
2040 PR(2,2)=6

```

```

2050 RETURN
2060 PRINT "The monster sleeps"
2070 RETURN
2080 PRINT "There is a snake in here"
2090 IF RND<.4 THEN RETURN
2100 IF IBMAX=1 OR RND<.5 THEN 2140
2110 IBMAX=IBMAX-1
2120 VP(B(IBMAX))=13+INT(RND*12)
2130 PRINT "The snake takes something "
2140 PRINT "It sneaks away"
2150 PR(3,1)=PR(3,1)+3
2160 IF PR(3,1)>24 THEN PR(3,1)=PR(3,1)-8
2170 RETURN
2180 ON PR(4,2)GOTO 2190,2220,2260
2190 PRINT "There is a giant hellhound in here. It looks like he wants you for dinner"
2200 PR(4,2)=2
2210 RETURN
2220 PRINT "The hellhound attacks you and bites you violently"
2230 CWD=CWD+1
2240 PR(4,2)=3
2250 RETURN
2260 PRINT "The hellhound may attack you again"
2270 PR(4,2)=2+INT(RND*2)
2280 RETURN
2290 PRINT "The nullity bomb is here"
2300 PRINT "There are three wires: a green one(g), a red one(r) and a yellow one(y)"
2310 PRINT "You must disconnect two of them"
2320 INPUT "Which will be the first one:":X$
2330 INPUT "And which will be the second one:":Y$
2340 CBO=0 :: C$="ryg"
2350 FOR I=1 TO 3
2360 IF X$=SEG$(C$,I,1)THEN CBO=CBO+1
2370 IF Y$=SEG$(C$,I,1)THEN CBO=CBO+1
2380 NEXT I
2390 IF CBO<2 THEN PRINT "Watch out, wrong input" :: GOTO 2300
2400 IF (X$&Y$)="yr" OR (X$&Y$)="ry" THEN 2440
2410 CALL CLEAR
2420 PRINT "By disconnecting the wrong wires, you set the bomb off and the earth was destroyed"

```



```

2430 END
2440 CALL CLEAR
2450 PRINT "Congratulations, you          succeeded where others          failed"
2460 END
2470 IF IVW=0 THEN 2520
2480 PR(6,2)=2
2490 IF VP(2)=IKM AND VP(12)=IKM THEN IVW=8 :: VP(IVW)=IKM :: VP(2)=0 :: VP(12)=
0 :: GOTO 2520
2500 VP(IVW)=0
2510 IVW=0
2520 ON PR(6,2)GOTO 2530,2550,2580
2530 PRINT "There is an enormous          barbecue here with a large  fire under i
t"
2540 RETURN
2550 PRINT "An awful smell fills your    nose"
2560 PR(6,2)=3
2570 RETURN
2580 PRINT "Everything is quiet now.      Even the terrible smell      fades"
2590 PR(6,2)=1
2600 RETURN
2610 REM
2620 IBZ=0
2630 FOR I=1 TO IBMAX-1
2640 IF B(I)=IVWB THEN IBZ=I :: I=IBMAX-1
2650 NEXT I
2660 RETURN
2670 REM
2680 CRO=CRO+1
2690 IF IKM=D(CRO)THEN CRI=CRI+1
2700 IF CRO<B THEN RETURN
2710 IF CRI=8 THEN 2770
2720 PRINT "Strange forces did somethingto you. It is dark"
2730 PRINT "For a moment you are          unconscious"
2740 CRO=1 :: CRI=1
2750 IKM=1
2760 RETURN
2770 PRINT "You hear a strange sound      like something being pushed away"
2780 PRINT "Now it has stopped"
2790 K(2,1)=3
2800 RETURN
2810 REM

```



```
2820 REM
2830 FOR I=1 TO 36
2840 K(I,1)=I+1 :: K(I,2)=I-1
2850 K(I,3)=I+4 :: K(I,4)=I-4
2860 NEXT I
2870 FOR I=0 TO 24 STEP 12
2880 FOR J=1 TO 9 STEP 4
2890 K(I+J+3,1)=0 :: K(I+J,2)=0
2900 NEXT J
2910 FOR J=1 TO 4
2920 K(I+J+8,3)=0 :: K(I+J,4)=0
2930 NEXT J
2940 NEXT I
```

```

2950 K(1,5)=16 :: K(7,5)=15
2960 K(32,6)=13 :: K(13,5)=32
2970 K(35,6)=18 :: K(18,5)=35
2980 FOR I=1 TO 15
2990 READ IKM :: READ IHD
3000 K(IKM,IHD)=0
3010 NEXT I
3020 REM
3030 IKM=36
3040 IBZ=1 :: IBMAX=1
3050 CKM=0
3060 CLL=0
3070 BLL=0
3080 CWD=0
3090 CRO=0 :: CRI=0
3100 REM
3110 FOR I=1 TO 15
3120 READ H$(I)
3130 NEXT I
3140 IN=-1
3150 FOR I=1 TO 12
3160 READ T$
3170 GOSUB 3520
3180 V$(I)=TN$
3190 NEXT I
3200 FOR I=1 TO 6
3210 READ T$
3220 GOSUB 3520
3230 P$(I)=TN$
3240 NEXT I
3250 FOR I=1 TO 36
3260 READ T$
3270 GOSUB 3520
3280 R$(I)=TN$
3290 NEXT I
3300 FOR I=1 TO 12
3310 READ VP(I)
3320 NEXT I
3330 FOR I=1 TO 6
3340 READ PR(I,1):: PR(I,2)=1
3350 READ PR(I,3)

```

```

3360 NEXT I
3370 FOR I=1 TO 8
3380 READ D(I)
3390 NEXT I
3400 IN=0
3410 RETURN
3420 DATA 21,1,22,2,22,1,23,2,18,1,19,2,16,3,20,4,11,1,12,2,7,1,8,2,7,4,3,3,2,1
3430 DATA east,west,north,south,up,down,kill,help,take,drop,invent,bandage  te
rnon,lanterno,read
3440 DATA mboufso,xifbuqjmf,xbufstbdl,dpplcppl,mfbgmmfu,cboebhft,txpse,dppljt,JU
(t!cpez,hjbmcpez,toblfcpez,efbe!ipvoe
3450 DATA J/U/,npotufs,toblif,ifmmipvoe,cbsefddf,cpnc
3460 DATA sftfudbwf,udbwf,tfdsfudpssejps,dpouspmsepn,pdbwf,jdbwf,tqbdfdbwf,cmbdl
sepn,qdbwf,fdbwf,odbwf,fnquzofst
3470 DATA tnbmmdbwf,spdlzdbwf,tnfmmzdbwf,esbhpodbwf,toblfdbwf,zfmpxdbwf,tusfbnc
bol,tu,jolzqmbdf,sfedbwf,gjobmdbwf
3480 DATA dpmpsdbwf,jdfdbwf,pqfoqmbdf,xppet,xppet,xppet,xppet,xppet,xppet,xppet,
xppet,xppet,xppet,xppet
3490 DATA 34,30,28,21,14,15,13,0,0,0,0,0
3500 DATA 34,2,16,15,17,4,29,2,8,1,25,1
3510 DATA 1,5,9,10,11,7,6,2
3520 REM
3530 TN$=""
3540 FOR I1=1 TO LEN(T$)
3550 C$=SEG$(T$,I1,1)
3560 C$=CHR$(ASC(C$)-1)
3570 TN$=TN$&C$
3580 NEXT I1
3590 IF IN=0 THEN PRINT TN$
3600 RETURN

```

# The Wolf and the Five Little Goats

A Grimm's fairy tale? No, an intriguing board game! This game is played on a checkerboard displayed on the screen. As the game begins, the goats are scattered throughout the lower half of the board and are represented by little squares containing a number. The wolf stands at the upper left-hand corner of the board.

Off we go — you are the wolf, and the computer controls the goats. You win if you eat three goats, and the computer wins if one of the goats eats you. (These wolf-eating goats are amazing creatures!)

You may start. On the lower part of the screen you see:

YOU MAY ENTER . . . MOVES

for example

YOU MAY ENTER 2 MOVES

The number of moves always lies between 1 and 3, and tells you how many steps you may take during your turn. The wolf — in other words, you — may move horizontally or vertically, but never diagonally. Each move must be entered on the cursor keys. If the *last* move of a series of moves brings you on a square with a goat, the goat is yours. You may never cross a square that has a goat on it.

There are 5 goats in all, and they may jump over each other. Goats can move only in one direction. The number of steps they may move is shown on the goats themselves, and is always between 1 and 5. For instance, if a goat bears the number 3 he can move

3 steps to the left, or  
3 steps to the right, or  
3 steps forward, or  
3 steps backward

He cannot move 1 step forward and 2 steps to the left. If you (the wolf) have moved to a new position the computer shows

NOW IT'S MY TURN

When it is your turn, the computer shows

YOU MAY ENTER . . . MOVES

At the end of the game the computer either tells you

YOU WIN

or worse

YOU LOSE

```
10 REM TI(EXTENDED)-THE WOLF AND THE 5 LITTLE GOATS
20 CALL CLEAR
30 RANDOMIZE
40 S1$="YOU MAY ENTER  MOVES"
50 S2$="NOW IT'S MY TURN"
60 GOTO 810
70 REM WOLF'S MOVE*****
80 DISPLAY AT(20,2):S1$
90 CALL HCHAR(20,18,BW+48)
100 CALL KEY(1,KE,ST)
110 IF ST=0 OR ST=-1 THEN 100
```

```

120 IF KE=0 THEN RWN=RW+1 :: KWN=KW :: GOTO 160
130 IF KE=2 THEN RWN=RW :: KWN=KW-1 :: GOTO 160
140 IF KE=5 THEN RWN=RW-1 :: KWN=KW :: GOTO 160
150 IF KE=3 THEN RWN=RW :: KWN=KW+1 ELSE 100
160 IF RWN<RBMIN OR RWN>RBMAX OR KWN<KBMIN OR KWN>KBMAX THEN 100
170 CALL GCHAR(RWN,KWN,GC)
180 IF GC>48 AND GC<54 AND BW<>1 THEN 100
190 BW=BW-1 :: CALL HCHAR(20,18,BW+48)
200 CALL HCHAR(RW,KW,30):: KW=KWN :: RW=RWN
210 CALL HCHAR(RW,KW,ASC("W"))
220 IF BW>0 THEN 100
230 FOR I=1 TO 5
240 IF RW<>P(I,1)OR KW<>P(I,2)THEN 260
250 IF P(I,3)<>0 THEN P(I,3)=0 :: BEAT=BEAT+1
260 NEXT I
270 BW=INT(RND*3+1)
280 RETURN
290 REM GOATS' MOVE*****
300 DISPLAY AT(20,2):S2$
310 MMAX=-400
320 FOR I=1 TO 5
330 IF P(I,3)=0 THEN 420
340 KS=P(I,2):: RS=P(I,1)-P(I,3)
350 GOSUB 500
360 RS=P(I,1)+P(I,3)
370 GOSUB 500
380 RS=P(I,1):: KS=P(I,2)+P(I,3)
390 GOSUB 500
400 KS=P(I,2)-P(I,3)
410 GOSUB 500
420 NEXT I
430 CALL HCHAR(P(IMAX,1),P(IMAX,2),30)
440 P(IMAX,1)=RMAX
450 P(IMAX,2)=KMAX
460 P(IMAX,3)=INT(RND*5+1)
470 CALL HCHAR(RMAX,KMAX,ASC(STR$(P(IMAX,3))))
480 RETURN
490 REM EVALUATION*****
500 STAT=0
510 IF RS<RBMIN OR RS>RBMAX OR KS<KBMIN OR KS>KBMAX THEN RETURN
520 D=ABS(RW-RS)*ABS(KW-KS)

```

```

530 IF D=BW THEN RETURN
540 IF D=BW-2 THEN STAT=-200
550 IF D=0 THEN STAT=500 :: GOTO 630
560 IF D<=5 THEN STAT=STAT+35
570 IF RS=RW THEN STAT=STAT+40
580 IF KS=KW THEN STAT=STAT+40
590 IF ABS(P(I,1)-RW)+ABS(P(I,2)-KW)=BW THEN STAT=STAT+60
600 FOR T=1 TO 5
610 STAT=STAT-((RS<>P(T,1))+(KS<>P(T,2)))*10
620 NEXT T
630 IF STAT+RND<MMAX THEN RETURN
640 MMAX=STAT :: IMAX=I :: RMAX=RS :: KMAX=KS
650 RETURN
660 REM INITIALIZE AND DRAW CHECKERBOARD*****
670 RBMIN=7 :: RBMAX=16 :: KBMIN=11 :: KBMAX=20
680 BW=2 :: MMAX=-400 :: BEAT=0
690 RW=RBMIN :: KW=KBMIN
700 FOR I=0 TO 9
710 CALL HCHAR(RBMIN+I,KBMIN,30,10)
720 NEXT I

```





```

730 FOR I=1 TO 5
740 P(I,1)=INT(RND*5)+RBMIN+5
750 P(I,2)=INT(RND*10)+KBMIN
760 P(I,3)=INT(RND*5)+1
770 CALL HCHAR(P(I,1),P(I,2),ASC(STR$(P(I,3))))
780 NEXT I
790 CALL HCHAR(RW,KW,ASC("W"))
800 RETURN
810 REM MAIN PROGRAM*****
820 GOSUB 670
830 GOSUB 80
840 IF BEAT=3 THEN PRINT "You win" :: END
850 GOSUB 300
860 IF MMAX<400 THEN 830
870 PRINT "You lose"
880 END

```

# Highway

This game shows the power of working with sprites. After the RUN command has been entered you will see a busy highway with many cars on it — your task is to get across safely. You use the joystick to control your movements (the ALPHA LOCK key must not be depressed otherwise the joystick will not work correctly).

At the start of the game the computer requests a level which determines the speed of the cars. The cars do not all travel at the same rate, however, and this makes your task all the more difficult.

If you are unlucky enough to be involved in an accident the computer asks

AGAIN?

and by pressing ENTER you are miraculously restored to health, and the game starts again.

```
10 REM TI(EXTENDED)-HIGHWAY
20 CALL CLEAR
30 INPUT "LEVEL ( <25 )=" :L
40 CALL CLEAR
50 S=0
60 DISPLAY AT(4,3):"LEVEL=";L
70 CALL SCREEN(16)
80 CALL COLOR(9,9,9)
90 CALL HCHAR(11,1,97,32)
100 CALL HCHAR(20,1,97,32)
110 CALL CHAR(42,"00FBABFBFFFF6363")
120 A$="18107EBDBD3C2424"
130 CALL CHAR(90,A$)
140 FOR K=1 TO 8
150 R=INT(RND*30+10)
160 CALL SPRITE(%K,42,K+3,81+8*K,8,0,R+L)
170 NEXT K
```

```

180 CALL SPRITE(#20,90,2,170,80+K*8)
190 CALL JOYST(1,X,Y)
200 CALL MOTION(#20,-3*Y,3*X)
210 CALL COINC(ALL,M)
220 IF M=-1 THEN GOTO 260
230 CALL POSITION(#20,YL,XL)
240 IF YL<85 THEN 330
250 GOTO 190
260 CALL MOTION(#20,0,0)
270 CALL SOUND(2000,-7,5)
280 DISPLAY AT(8,8):"AGAIN?"
290 ACCEPT AT(8,16):Q$
300 IF SEG$(Q$,1,1)="N" THEN 370
310 CALL DELSPRITE(ALL)
320 GOTO 20
330 S=S+1
340 DISPLAY AT(5,3):"SCORE=";S
350 CALL DELSPRITE(#20)
360 GOTO 180
370 END

```

# Road Race

Ladies and gentlemen, the cars are now on the grid waiting for the start of the most thrilling Grand Prix you are ever likely to see. With the world championship at stake none of these drivers will be looking for anything other than first place. The weather is fine and the track is dry so it's sure to be a very fast race. Who knows, we may even see the lap record broken. The last minute checks have been made, the drivers are now revving their engines up to full power and, as the starter's flag drops, they're off . . . !

### After the RUN command the computer asks

## ONE OR TWO PLAYERS?

Then it asks if you want to play with joysticks. If you don't, you control your cars with the following keys.

```

      E      I
S      D      J      K
      X      M
    car1    car2

```

**Finally, it asks the number of laps you want the race to be.**

**You will soon find out that the cars are subject to the same physical laws as in the real world. This makes the game slightly harder than you might expect. All we can suggest is practice, practice, and practice again!**

```

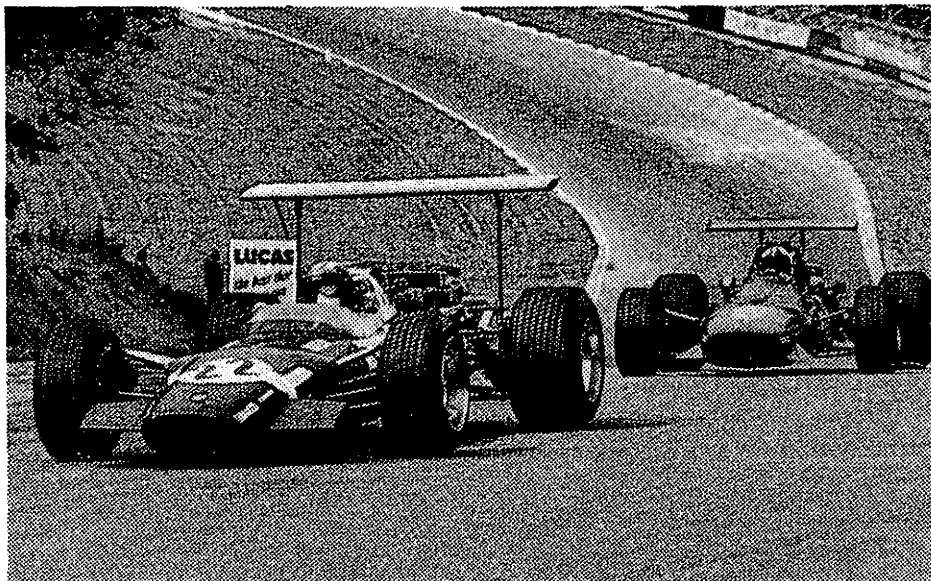
10 REM TI(EXTENDED)-ROADRACE
20 CALL CLEAR
30 OPTION BASE 1
40 DIM CL(2),FT(2),DT(2),RO(2),MT(2),LT(2),XOLD(2),YOLD(2),XNEW(2),YNEW(2),VX(2)
,VY(2),ACCX(2),ACCY(2),CF(2),SO(2),RE(1)
50 LA=2 :: RA=3 :: UA=5 :: DA=0
60 FR=.9 :: CL(1)=9 :: CL(2)=5 :: F1=50/60 :: F2=60*F1 :: RE(1)=300
70 GOTO 1320
80 REM **INITIALISATION
90 CALL CHAR(96,"E047427F7F4247E0")
100 CALL CHAR(97,"07E242FEFE42E207")
110 CALL CHAR(98,"5A7E5A181899FFB1")
120 CALL CHAR(99,"81FF9918185A7E5A")
130 CALL CHAR(33,"FFFEFCF8F0E0C080")
140 CALL CHAR(34,"0103070F1F3F7FFF")
150 CALL CHAR(35,"FF7F3F1F0F070301")
160 CALL CHAR(36,"80C0E0F0F8FCFEFF")
170 CALL CHAR(37,"FFFFFFFFFFFFFFFF")
180 CALL CHAR(62,"0000000402FF0204")
190 CALL CLEAR :: CALL SCREEN(13):: CALL COLOR(1,1,1):: CALL HCHAR(1,1,37,768)
200 CALL COLOR(1,1,12,2,2,12)
210 READ X,PX,PY :: FOR I=1 TO X
220 READ DIR,LN :: IF DIR=0 THEN 240
230 ON DIR GOTO 250,260,270,280,290,300,310,320
240 PX=PX+LN :: READ EXTRA :: PY=PY+EXTRA :: GOTO 220
250 CALL HORIZONTAL(PX,PY,LN):: PX=PX+LN :: GOTO 360
260 PX=PX+LN :: PY=PY-LN :: CALL DIAGONAL(PX-1,PY,-1,LN):: GOTO 360
270 PY=PY-LN :: CALL VERTICAL(PX,PY,LN):: GOTO 360
280 PX=PX-LN :: PY=PY-LN :: CALL DIAGONAL(PX,PY,1,LN):: GOTO 360
290 PX=PX-LN :: CALL HORIZONTAL(PX,PY,LN):: GOTO 360
300 CALL DIAGONAL(PX-1,PY,-1,LN):: PX=PX-LN :: PY=PY+LN :: GOTO 360
310 CALL VERTICAL(PX,PY,LN):: PY=PY+LN-3 :: GOTO 360
320 CALL DIAGONAL(PX,PY,1,LN):: PX=PX+LN :: PY=PY+LN :: GOTO 360
330 DATA 19,7,1
340 DATA 1,8,8,3,1,1,2,3,1,2,8,7,0,-3,0,7,12,0,3,0,6,5,5,11
350 DATA 3,10,0,3,0,4,3,5,3,6,3,7,9,0,3,0,6,4,5,1,4,4,0,0,3,3,16,2,4
360 NEXT I
370 CALL CHAR(62,"0000000402FF0204"):: CALL HCHAR(4,8,62)
380 CALL CHAR(45,"0030300000303000"):: CALL VCHAR(1,8,45,3)
390 RETURN
400 REM **START

```

```

410 DISPLAY ERASE ALL AT(3,3):"ONE OR TWO PLAYERS? 2"
420 ACCEPT AT(3,24)SIZE(-1)BEEP VALIDATE("12"):NP
430 DISPLAY AT(5,3):"JOYSTICKS? Y" :: ACCEPT AT(5,15)BEEP SIZE(-1)VALIDATE("Yn")
440 JO=(AN$="Y")OR(AN$="y")
450 DISPLAY AT(7,3):"NUMBER OF LAPS? 1" :: ACCEPT AT(7,22)BEEP SIZE(-1)VALIDATE
(DIGIT):NR
460 NR=INT(NR):: IF NR<1 THEN 450
470 FOR PL=1 TO 2
480 XNEW(PL)=54 :: VX(PL)=0 :: XOLD(PL)=54
490 YNEW(PL)=12*PL-4 :: VY(PL)=0 :: YOLD(PL)=YNEW(PL)
500 DT(PL)=0 :: MT(PL)=1E6 :: LT(PL)=0 :: RO(PL)=0
510 SO(PL)=40000 :: CF(PL)=1 :: NEXT PL
520 NF=0 :: MI=0
530 DC=1 :: RETURN
540 FOR PL=1 TO NP
550 CALL SPRITE(#PL,96,CL(PL),YNEW(PL)-4,XNEW(PL)-4)
560 NEXT PL
570 RETURN
580 REM **INPUT ACCELERATION
590 IF JO THEN 660
600 ACCX(PL)=0 :: ACCY(PL)=0 :: CALL KEY(PL,K,X)
610 IF K=LA THEN ACCX(PL)=-4
620 IF K=RA THEN ACCX(PL)=4
630 IF K=UA THEN ACCY(PL)=4
640 IF K=DA THEN ACCY(PL)=-4
650 RETURN
660 CALL JOYST(PL,ACCX(PL),ACCY(PL))
670 RETURN
680 REM **CALCULATE NEW SPEED & PLACE
690 VX(PL)=(VX(PL)+ACCX(PL))*FR
700 VY(PL)=(VY(PL)-ACCY(PL))*FR
710 XOLD(PL)=XNEW(PL):: XNEW(PL)=XNEW(PL)+VX(PL)
720 YOLD(PL)=YNEW(PL):: YNEW(PL)=YNEW(PL)+VY(PL)
730 RETURN
740 REM **MOVE
750 XINT=INT(XNEW(PL)): YINT=INT(YNEW(PL)): IF XINT<5 OR YINT<5 OR XINT>255 OR
YINT>188 THEN 820
760 CALL GCHAR(INT(YINT/8+1),INT(XINT/8+1),CH):: IF CH=32 OR CH=45 THEN 850
770 IF CH>36 THEN 820
780 CALL CHARPAT(CH,CHAR$)

```



```

CTL$=SEG$(CHAR$,INT((YINT/8-INT(YINT/8))*16+1),2)
CALL HEXBIN(CTL$,BIN$)
IF SEG$(BIN$, (XINT/8-INT(XINT/8))*8+1,1)="0" THEN 850
XNEW(PL)=XOLD(PL):: YNEW(PL)=YOLD(PL)
DT(PL)=INT(ABS(VX(PL))+ABS(VY(PL)))
CALL MOTION(#PL,0,0):: VX(PL)=0 :: VY(PL)=0 :: RETURN
IF ABS(VY(PL))>ABS(VX(PL))THEN 890
IF VX(PL)<0 THEN CVAL=97 :: GOTO 910
IF VX(PL)>0 THEN CVAL=96 :: GOTO 910
GOTO 920
IF VY(PL)<0 THEN CVAL=98 :: GOTO 910
CVAL=99
CALL PATTERN(#PL,CVAL)
CALL LOCATE(#PL,YNEW(PL)-4,XNEW(PL)-4)
SO(PL)=10*(ABS(VX(PL))+ABS(VY(PL)))+109 :: IF SO(PL)<110 THEN SO(PL)=40000

```

```

940 CALL MOTION(#PL,VY(PL)/5,VX(PL)/5)
950 RETURN
960 REM **FINISHED?
970 TIMER=TIMER+1 :: TI=TIMER/F1+MI
980 IF TI-MI>F2 THEN TIMER=TIMER-F2 :: MI=MI+60
990 FOR PL=1 TO 2
1000 ON CF(PL)GOTO 1010,1030,1050,1070
1010 IF XNEW(PL)>200 THEN CF(PL)=2 :: GOTO 1130
1020 GOTO 1130
1030 IF YNEW(PL)>150 AND XNEW(PL)<150 THEN CF(PL)=3 :: GOTO 1130
1040 GOTO 1130
1050 IF XNEW(PL)<50 THEN CF(PL)=4 :: GOTO 1130
1060 GOTO 1130
1070 IF NOT(XNEW(PL)>50 AND YNEW(PL)<30)THEN 1130
1080 CF(PL)=1 :: RT=TI-LT(PL):: LT(PL)=TI
1090 IF RT<MT(PL)THEN MT(PL)=RT
1100 RO(PL)=RO(PL)+1 :: IF RO(PL)<NR THEN 1130
1110 NF=NF+1 :: FT(PL)=TI
1120 FOR S1=1 TO 3 :: FOR S2=200 TO 700 STEP 50 :: CALL SOUND(-99,S2,0):: NEXT S
2 :: NEXT S1
1130 NEXT PL
1140 RETURN
1150 REM **END
1160 CALL CLEAR :: CALL CHARSET :: CALL DELSPRITE(ALL):: CALL SCREEN(11)
1170 PRINT "RECORD TIME:";:: TI=RE(DC):: GOSUB 1290
1180 PRINT :: PRINT "NUMBER OF LAPS: ";NR :: PRINT
1190 FOR PL=1 TO NP
1200 PRINT :: PRINT
1210 PRINT "PLAYER ";PL :: PRINT
1220 PRINT "TOTAL: ";:: TI=FT(PL):: GOSUB 1290
1230 PRINT "AVERAGE:";:: TI=FT(PL)/RO(PL):: GOSUB 1290
1240 PRINT "FASTEST:";:: TI=MT(PL):: GOSUB 1290
1250 IF TI<RE(DC)THEN RE(DC)=TI :: PRINT "NEW TRACK RECORD!!!" :: PRINT
1260 NEXT PL
1270 RETURN
1280 REM **OUTPUT TIME
1290 MI=INT(TI/60):: PRINT USING " ##:##.##":MI,TI-60*MI
1300 RETURN
1310 REM **MAIN PROGRAM**
1320 GOSUB 410
1330 RESTORE :: GOSUB 90 :: GOSUB 540

```



```

1340 FOR PL=1 TO 2
1350 IF PL>NP THEN CALL SOUND(-50,SO(1),0):: CALL SOUND(1000,SO(1),1):: GOTO 1400
1360 IF DT(PL)>0 THEN DT(PL)=DT(PL)-1 :: SO(PL)=-6 :: GOTO 1400
1370 GOSUB 590
1380 GOSUB 690
1390 GOSUB 750
1400 NEXT PL
1410 IF SO(1)<>SO(2) THEN CALL SOUND(-4000,SO(1),1,SO(2),1):: GOTO 1430
1420 CALL SOUND(-4000,SO(1),1)
1430 GOSUB 970
1440 IF NF<NP THEN 1340
1450 GOSUB 1160
1460 PRINT :: INPUT "PLAY AGAIN (Y/N)? ":ANS
1470 AN$=SEG$(AN$,1,1):: IF AN$="Y" OR AN$="y" THEN 1320
1480 END
1490 SUB HORIZONTAL(STX,STY,LENGHT)
1500 FOR I=0 TO 2
1510 CALL HCHAR(STY+I,STX,32,LENGHT)
1520 NEXT I
1530 SUBEND
1540 SUB VERTICAL(STX,STY,LENGHT)
1550 FOR I=0 TO 2
1560 CALL VCHAR(STY,STX+I,32,LENGHT)
1570 NEXT I
1580 SUBEND
1590 SUB DIAGONAL(STX,STY,DX,LENGHT)
1600 IF DX=1 THEN CH1=35 :: CH2=36 :: GOTO 1630
1610 IF DX=-1 THEN CH1=33 :: CH2=34 :: GOTO 1630
1620 SUBEXIT
1630 FOR I=0 TO LENGHT-1
1640 X=STX+DX*I :: Y=STY+I
1650 CALL GCHAR(Y,X,CH):: IF CH=37 THEN CALL HCHAR(Y,X,CH1)
1660 CALL VCHAR(Y+1,X,32,2)
1670 CALL GCHAR(Y+3,X,CH):: IF CH=37 THEN CALL HCHAR(Y+3,X,CH2)
1680 NEXT I
1690 SUBEND
1700 SUB HEXBIN(HEX$,BIN$)
1710 BIN$=""
1720 FOR I=LEN(HEX$) TO 1 STEP -1
1730 A$=SEG$(HEX$,I,1):: IF ASC(A$)>64 THEN 1750

```

```
1740 DEC=VAL(A$):: GOTO 1760
1750 DEC=ASC(A$)-55
1760 FOR J=1 TO 4
1770 N=INT(DEC/2):: BIN$=STR$(INT(DEC-2*N+.1))&BIN$
1780 DEC=N :: NEXT J
1790 NEXT I
1800 SUBEND
```

# At the Market

Have you ever wandered through a market and been amazed at the speed with which the salesmen can add up a list of prices? Play this game with your family and find out how good a market trader you would make.

You will see pairs of numbers of increasing length which you have to add up. You will soon find out that this isn't as easy as it sounds. To see why, consider the sum

$$\begin{array}{r} 75856 \\ + 37637 \\ \hline \end{array}$$

Normally you would add the numbers in the right hand column first, then the column next to it, and so on. With the computer, however, you must enter the answer starting with the left hand column. It is this difference that makes the game so tricky.



How many numbers can you add correctly within the time limit of about one minute? Can you beat our record of eight?

```
10 REM TI-AT THE MARKET
20 RANDOMIZE
30 DIM JO$(9)
40 FOR I=0 TO 9
50 READ JO$(I)
60 NEXT I
70 CALL CLEAR
80 NR=0
90 TI=0
100 MX=1
110 GOTO 600
120 DATA ZERO,SCHOOL-CHILD,PAPER-BOY,CLERK,AUTO-DEALER,BROKER
130 DATA DIRECTOR,OIL MAGNATE,MILLIONAIRE,WORLD-CHAMPION
140 REM**RANDOM ADDITION**
150 N1$=STR$(INT(RND*MX))
160 N2$=STR$(INT(RND*MX))
170 AN=VAL(N1$)+VAL(N2$)
180 L1=LEN(N1$)
190 L2=LEN(N2$)
200 D=L1-L2
210 FOR I=1 TO ABS(D)
220 IF L1=L2 THEN 270
230 IF L2<L1 THEN 260
240 N1$=" "&N1$
250 GOTO 270
260 N2$=" "&N2$
270 NEXT I
280 PRINT N1$
290 PRINT N2$;"+"
300 FOR DU=0 TO NR+1
310 PRINT "-";
320 NEXT DU
330 RETURN
340 REM**INPUT ANSWER**
350 AN$=""
360 PRINT "ANSWER? ";
370 TI=TI+1
```

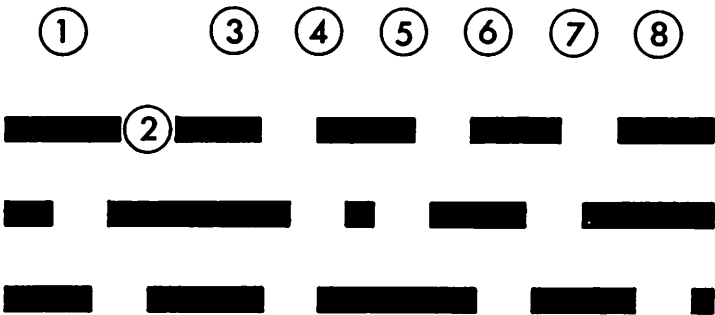
```

380 IF TI=1500 THEN 530
390 CALL KEY(3,DI,S)
400 IF S=0 THEN 370
410 IF DI<>8 THEN 450
420 IF AN$="" THEN 370
430 PRINT "/"&SEG$(AN$,LEN(AN$),LEN(AN$))&"/";
440 GOTO 490
450 PRINT CHR$(DI);
460 IF DI<48 THEN 490
470 IF DI>57 THEN 370
480 AN$=AN$&CHR$(DI)
490 IF DI<>8 THEN 510
500 AN$=SEG$(AN$,1,LEN(AN$)-1)
510 IF DI=13 THEN 540
520 GOTO 370
530 PRINT
540 RETURN
550 REM**CONTROL ANSWER**
560 IF AN$<>" THEN 580
570 AN$="-1"
580 IF VAL(AN$)=AN THEN 700 ELSE 740
590 RETURN
600 REM**MAIN PROGRAM**
610 MX=MX+10
620 GOSUB 140
630 GOSUB 340
640 GOSUB 550
650 IF VAL(AN$)<>AN THEN 670
660 IF NR<10 THEN 610
670 PRINT "JOB SPECIFICATION: "
680 PRINT JO$(NR)
690 END
700 REM**RIGHT ANSWER**
710 PRINT "RIGHT!"
720 NR=NR+1
730 GOTO 590
740 REM**WRONG ANSWER**
750 PRINT "WRONG!"
760 PRINT "IT IS: ";AN
770 GOTO 590

```

# Fallout

At the start of this simple but absorbing game you will see eight horizontal bars with gaps in them. Above the bars are eight checkers which can fall through the gaps. In this example



checker 2 has already fallen into a gap. The object of the game is to get all eight checkers through the bars. You can do this by moving the bars to line up the gaps for the checkers to fall through.

To move a particular bar enter a command of the form

**BDS**

where

B is the number of the bar (the bars are numbered from 1 at the top to 8 at the bottom)

D is the direction you wish to move it (L for left, R for right) and

S is the number of steps the bar has to be shifted (up to a maximum of 9).

For instance

3R2

moves bar 3, 2 steps to the right. This might sound like a very simple game but when you actually start playing it you will find that it provides quite a stiff test of your ability to think logically.

```
10 REM TI(EXTENDED)-FALL OUT
20 DIM MAT(9,18),CHT(9,18)
30 GOSUB 550
40 ROW=1
50 GOSUB 300
60 GOSUB 430
70 REM MAIN PROGRAM*****
80 GOSUB 690
90 FOR CNT=1 TO AZ
100 IF RI=-1 THEN 200
110 FOR CNT2=17 TO 1 STEP -1
120 CHT(ROW,CNT2+1)=CHT(ROW,CNT2)
130 MAT(ROW,CNT2+1)=MAT(ROW,CNT2)
140 NEXT CNT2
150 CHT(ROW,1)=CHT(ROW,18)
160 MAT(ROW,1)=MAT(ROW,18)
170 GOSUB 300
180 GOSUB 430
190 GOTO 280
200 FOR CNT2=1 TO 17
210 CHT(ROW,CNT2-1)=CHT(ROW,CNT2)
220 MAT(ROW,CNT2-1)=MAT(ROW,CNT2)
230 NEXT CNT2
240 CHT(ROW,17)=CHT(ROW,0)
250 MAT(ROW,17)=MAT(ROW,0)
```



```

260 GOSUB 300
270 GOSUB 430
280 NEXT CNT
290 GOTO 80
300 REM PLACE CHECKERS*****
310 FOR C1=1 TO 17
320 IF MAT(ROW,C1)=-1 THEN GOSUB 360
330 NEXT C1
340 RETURN
350 REM DROP CHECKERS*****
360 OG=9
370 FOR C2=8 TO 1 STEP -1
380 IF MAT(C2,C1)=-1 AND CHT(C2,C1)=0 THEN 400
390 OG=C2-1 :: GOTO 410
400 IF CHT(C2-1,C1)>0 THEN CHT(OG,C1)=CHT(C2-1,C1):: CHT(C2-1,C1)=0 :: OG=OG-(OG
=9)-1
410 NEXT C2
420 RETURN
430 REM DRAW SCREEN*****
440 NT=NT+1
450 FOR C6=0 TO 8

```



```

460 FOR C7=1 TO 17
470 IF MAT(C6,C7)=0 THEN CALL HCHAR(5+C6,5+C7,128):: GOTO 500
480 IF CHT(C6,C7)<>0 THEN CALL HCHAR(5+C6,5+C7,CHT(C6,C7)+48):: GOTO 500
490 CALL HCHAR(5+C6,5+C7,31)
500 NEXT C7
510 CALL HCHAR(5+C6,3,C6+48)
520 NEXT C6
530 DISPLAY AT(23,18)BEEP:NT
540 RETURN
550 REM initialize*****
560 NT=-1
570 FOR CC=1 TO 9
580 CHT(0,2*CC-1)=CC :: MAT(0,2*CC-1)=-1 :: MAT(0,CC*2)=1
590 NEXT CC
600 CALL CHAR(128,"00FFFFFFFFFFFFFF")
610 CALL CLEAR
620 PRINT "NUMBER OF TURNS: "
630 FOR TT=1 TO 8
640 FOR TTT=1 TO 4
650 MAT(TT,INT(RND*17+1))=-1
660 NEXT TTT
670 NEXT TT
680 RETURN
690 REM input move*****
700 CALL HCHAR(18,4,63,3)
710 CALL SOUND(10,880,0)
720 FOR C3=1 TO 3
730 CALL KEY(J,K,S):: IF S=0 OR S=-1 THEN 730
740 A(C3)=K
750 CALL HCHAR(18,3+C3,K)
760 NEXT C3
770 ROW=A(1)-48 :: IF ROW<1 OR ROW>8 THEN 700
780 RI=(A(2)-76)-(A(2)-82)
790 IF RI=0 THEN 700
800 AZ=A(3)-48 :: IF AZ<1 OR AZ>9 THEN 700
810 RETURN
820 GOTO 700

```

# Ship's Attack

First, let's describe this game as realistically as possible. In the lower half of the screen is a shape like this:



Little squares fall down from the top of the screen. You must move the shape using the cursor controls



to stop the squares hitting it. If that sounds dull what about this . . .

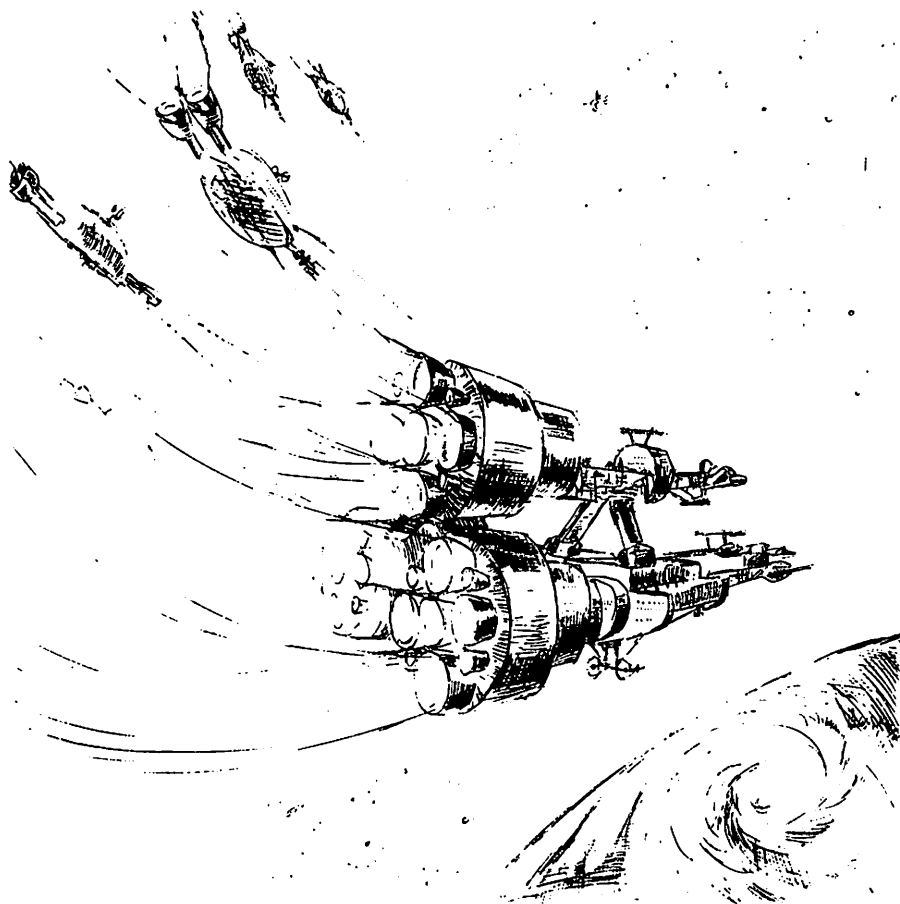
You are captain of one of the finest spaceships in the universe. As you cruise majestically through the Milky Way you can't help feeling proud of the magnificent vessel under your command. Then just as you are nearing base and the end of your voyage is in sight, disaster strikes —

*A FLYING SAUCER ATTACK!*

How can you escape these unfriendly invaders from across the galaxy? Why not try your

*SUPER ATOMIC ESCAPE MOTOR?*

That sounds better, doesn't it? Anyway, the game is fast, simple and great fun to play, and that is all that matters.



```

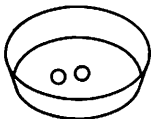
10 REM TI(EXTENDED)-SHIPS ATTACK
20 DIM R(16)
30 CALL CLEAR
40 INPUT "LEVEL (1-5)? ":LE
50 CALL MAGNIFY(2)
60 SC=0
70 CALL COLOR(3,16,1)
80 CALL COLOR(4,16,1)
90 A$="003C7EDB7E3C0000"
100 B$="00000000B1423C00"
110 CALL CHAR(42,A$)
120 CALL CHAR(95,B$)
130 CALL CLEAR
140 CALL SCREEN(2)
150 CALL SPRITE(#20,95,16,150,75)
160 FOR K=3 TO 16
170 R(K)=INT(RND*10+2*LE)
180 CALL SPRITE(#K,42,K,16,17*K-41,R(K),0)
190 GOSUB 410
200 NEXT K
210 FOR K=3 TO 16
220 GOSUB 410
230 IF M=-1 THEN K=16
240 SC=SC+1
250 R(K)=R(K)+LE
260 CALL MOTION(#K,R(K),0)
270 DISPLAY AT(3,3):SC
280 CALL SOUND(5,440,5)
290 NEXT K
300 IF M<>-1 THEN GOTO 210
310 CALL SOUND(2000,-7,5)
320 CALL SCREEN(12)
330 CALL DELSPRITE(ALL)
340 CALL COLOR(3,2,1)
350 CALL COLOR(4,2,1)
360 DISPLAY AT(20,2):"AGAIN (Y/N)?"
370 ACCEPT AT(20,15):Q$
380 IF SEG$(Q$,1,1)="Y" OR SEG$(Q$,1,1)="y" THEN GOTO 30
390 END
400 REM MOVE SHIP

```

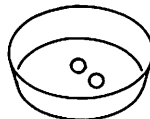
```
410 CALL JOYST(1,X,Y)
420 CALL MOTION(#20,0,3*X)
430 CALL COINC(ALL,M)
440 RETURN
```

# Mini Mancala

MINI MANCALA is based on an old Arabian game. It is played by moving stones between cups. There are four cups: A and B are the computer's, and C and D are yours. At the start of the game there are two stones in each cup.



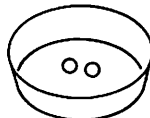
**A (computer)**



**B (computer)**

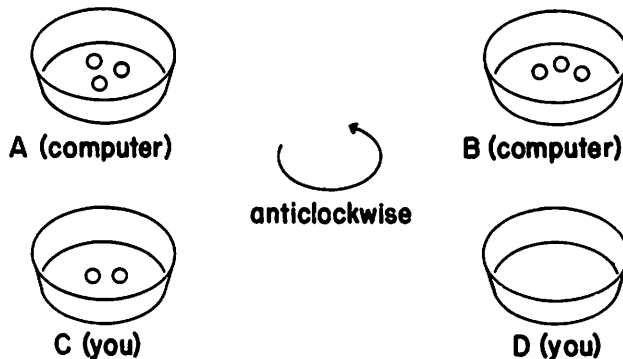


**C (you)**



**D (you)**

In turn, the players take the stones from one of their own cups and distribute them counterclockwise to the other three cups. For instance you might choose to move the stones from cup D like this:

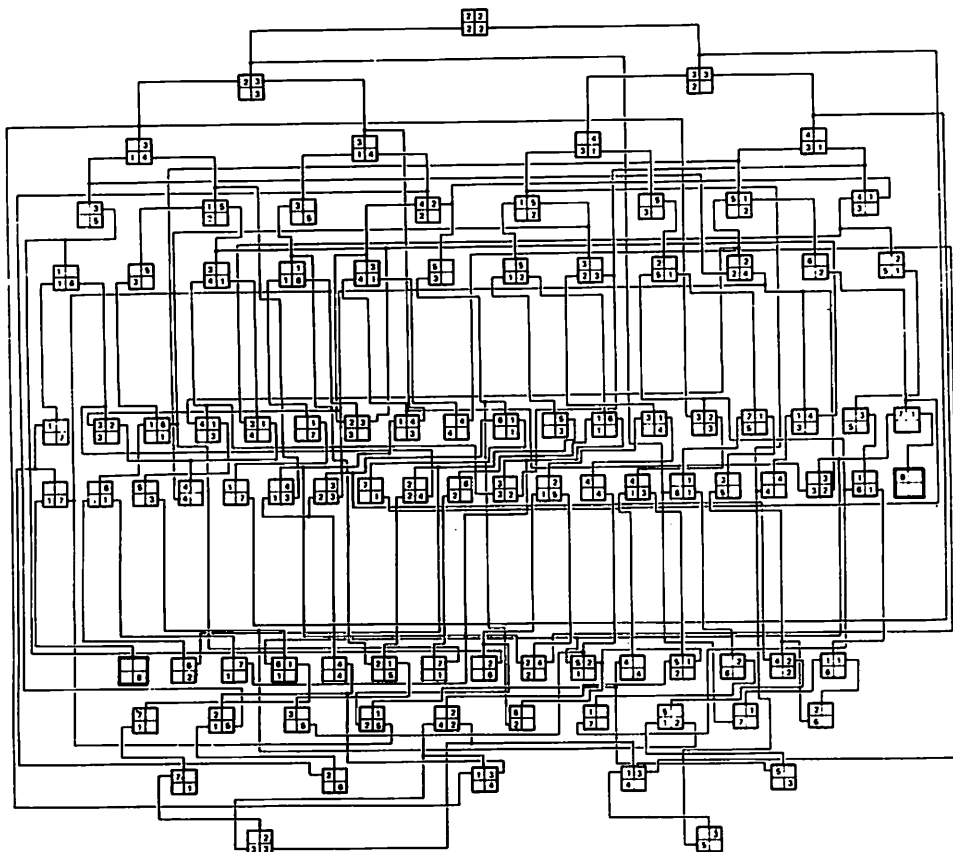


After this move cup D is empty. In fact, there will always be at least one empty cup, because during each move stones cannot be put back into the cup from which they were taken.

To win the game you must get all the stones into your own cups.

On the computer, the cups are represented by squares with numbers on them indicating the number of stones they contain. You can choose the level of difficulty you prefer, 1, 2, or 3 (1 being the easiest), and who has the first move. The computer will ask which of your squares you wish to move the stones from and will tell you what its own move is. The position of the stones on the board is displayed after each move.

You may be surprised that this complicated game can be described in such a short program. The diagram shows all the possible moves, and is an excellent example of how a strategic game can be represented schematically.





```

10 REM TI(EXTENDED)-MINI MANCALA
20 DIM M(1,1),L(3,1)
30 GOTO 400
40 REM INITIALIZE*****
50 FOR I=0 TO 1
60 M(I,1)=2
70 M(I,0)=2
80 FOR J=0 TO 3
90 READ L(J,I)
100 NEXT J
110 NEXT I
120 DATA 0,1,1,0,0,0,1,1
130 RETURN
140 REM SCREEN*****
150 PRINT
160 PRINT "A      B"
170 PRINT M(0,0);M(0,1)
180 PRINT
190 PRINT M(1,0);M(1,1)
200 PRINT "C      D"
210 RETURN
220 INPUT "MOVE STONES FROM CUP(C,D)? ":X$
230 IF (X$="D" OR X$="d")AND M(1,1)>0 THEN LB=2 :: RETURN
240 IF (X$="C" OR X$="c")AND M(1,0)>0 THEN LB=1 :: RETURN
250 PRINT "WRONG INPUT"
260 GOTO 220
270 RETURN
280 REM COMPUTERS TURN*****
290 PRINT "NOW IT'S MY TURN"
300 Q=M(0,0)*1000+M(0,1)*100+M(1,0)*10+M(1,1)
310 LB=3
320 IF M(0,0)=0 THEN 350
330 IF M(0,1)=0 THEN LB=0 :: GOTO 350
340 IF M(0,0)>0 AND (LEV<3 AND RND*LEV<.4)OR M(0,1)=6 OR Q=1430 OR Q=1340 OR Q=61
10 OR Q=1160 THEN LB=0
350 PRINT "I MOVE STONES FROM CUP ";CHR$(65-(LB=3))
360 RETURN
370 REM SPREAD*****
380 G=L(LB,0)
390 H=L(LB,1)
400 IF M(G,H)<>0 THEN 420

```

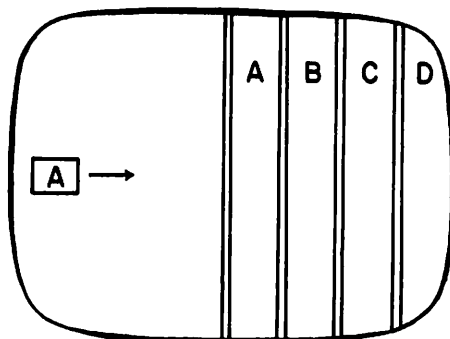
```

410 RETURN
420 LB=LB+1
430 IF LB>3 THEN LB=LB-4
440 I=L(LB,0):: J=L(LB,1)
450 M(G,H)=M(G,H)-1 :: M(I,J)=M(I,J)+1
460 GOTO 400
470 REM MAIN PROGRAM*****
480 GOSUB 50
490 GOSUB 150
500 INPUT "LEVEL(1-3) ":LEV
510 IF LEV<1 OR LEV>3 THEN 500
520 INPUT "WHO STARTS? YOU(Y) OR ME(M) ":X$
530 IF SEG$(X$,1,1)="M" OR SEG$(X$,1,1)="m" THEN 570
540 GOSUB 220
550 GOSUB 380
560 GOSUB 150
570 IF M(1,1)=8 THEN PRINT "YOU WIN" :: GOTO 630
580 GOSUB 290
590 GOSUB 380
600 GOSUB 150
610 IF M(0,0)=8 THEN PRINT "I WIN" :: GOTO 630
620 GOTO 540
630 INPUT "DO YOU WANT TO PLAY AGAIN ":X$
640 IF SEG$(X$,1,1)="Y" OR SEG$(X$,1,1)="y" THEN RESTORE :: GOTO 480
650 END

```

# Stop It!

Although this is only a short program it gives rise to a fast and exciting game. The screen looks like this



The square on the left has a letter on it. When it moves across the screen you must try and stop it, by pressing the S key, in the region bearing the same letter.

The computer will ask you how many times you want to play and will tell you at the end of the game how successful you were, for instance

NUMBER OF HITS: 0

Never mind, better luck next time!

```

10 REM TI-STOP IT
20 CALL CLEAR
30 INPUT "NUMBER OF TURNS?":B
40 CALL CLEAR
50 REM SCREEN*****
60 FOR A=1 TO 5
70 CALL VCHAR(1,19+2*A,30,24)
80 CALL HCHAR(1,20+2*A,96+A)
90 NEXT A
100 REM START GAME
110 I=INT(RND*5)
120 D=INT((RND*22)/2)*2+2
130 FOR A=4 TO 32 STEP 2
140 CALL HCHAR(D,A-2,31)
150 CALL KEY(1,KEY,STA)
160 IF KEY<>2 THEN 230
170 IF 24+2*I=A THEN 200
180 CALL SOUND(100,110,0)
190 GOTO 260
200 CALL SOUND(100,2000,0)
210 G=G+1
220 GOTO 260
230 CALL HCHAR(D,A,I+65)
240 NEXT A
250 REM NEXT TURN*****
260 T=T+1
270 IF B=T THEN 290
280 GOTO 110
290 PRINT "NUMBER OF TURNS:";B
300 PRINT "NUMBER OF HITS:";G

```

# BABA

A *thema con variatone* is a tune that, although it is based on one that has been heard before, has its own mood and identity. This game could be considered as a *thema con variatone* as it has some similarities with another game in this book, but presents its own unique challenge to the solver.

You will see 16 fields filled with a random arrangement of As and Bs e.g.

B	A	B	A
A	A	A	B
A	B	A	B
B	B	A	B

When you enter one of the fields (using the key shown on the screen) all the letters on the horizontal and vertical rows through that field will be altered so that all the As become Bs and vice versa. Your aim is to end up with a screen

which looks like this:

A	B	B	A
A	B	B	A
A	B	B	A
A	B	B	A

```
10 REM TI-SWEDISH POPSONG
20 CALL CLEAR
30 DIM A(16)
40 REM INITIALIZE*****
50 FOR K=1 TO 16
60 READ A(K)
70 NEXT K
80 RANDOMIZE
90 FOR K1=1 TO 9
100 R=INT(RND*4+1)
110 C=INT(RND*4+1)
120 GOSUB 230
130 NEXT K1
140 REM MAIN PROGRAM*****
150 GOSUB 430
160 INPUT "COLUMN=":C
170 IF (C<0)+(C>4)<>0 THEN 160
180 INPUT "ROW=":R
190 IF (R<0)+(R>4)<>0 THEN 180
200 GOSUB 230
210 GOSUB 430
220 GOTO 160
```



Courtesy Epic Records



```

230 REM DETERMINE*****
240 FOR H=1 TO 4
250 B(H)=(R-1)*4+H
260 B(H+4)=C+(H-1)*4
270 NEXT H
280 FOR K=1 TO 4
290 FOR J=K+1 TO 8
300 IF B(J)<>B(K)THEN 340
310 W=B(K)
320 B(K)=B(B)
330 B(B)=W
340 NEXT J
350 NEXT K
360 FOR J=1 TO 7
370 H=B(J)
380 A(H)=A(H)+1
390 IF A(H)<=1 THEN 410
400 A(H)=0
410 NEXT J
420 RETURN
430 REM DISPLAY*****
440 PRINT
450 PRINT "1234"
460 FOR K=1 TO 4
470 FOR J=1 TO 4
480 X$="A"

```



```
490 IF A(J+(K-1)*4)=0 THEN 510
500 X$="B"
510 PRINT X$;
520 NEXT J
530 PRINT K
540 NEXT K
550 PRINT
560 RETURN
570 DATA 0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0
```

# Vowels and Consonants

This competition game can be played by the whole family. Unlike most other computer games, it does not involve numbers or arithmetic. The computer will give you seven letters: when it asks you

## VOWEL OR CONSONANT (V/C)

enter V for a vowel or C for a consonant. When you have your seven letters the computer challenges you to make as long a word as possible out of them, using each letter once only. There is a time limit, which is shown on a special clock that 'counts down' through the alphabet. It is amazing how addictive a simple game like this can become. Try it and see!

```
10 REM TI-VOWELS AND CONSONANTS
20 RANDOMIZE
30 A1$="AEIOUY"
40 A2$="BCDFGHJKLMNPQRSTVWXZ"
50 CALL CLEAR
60 PRINT "Vowel or consonant(v/c): "
70 FOR CC=1 TO 8
80 CALL KEY(3,AW,S)
90 IF S=0 THEN 80
100 IF AW=86 THEN 130
110 IF AW=67 THEN 150
120 GOTO 80
130 C$=SEG$(A1$,INT(RND*6+1),1)
140 GOTO 160
150 C$=SEG$(A2$,INT(RND*20+1),1)
160 Z$=Z$&C$
170 CALL HCHAR(10,CC+2,ASC(C$))
180 NEXT CC
190 CALL CLEAR
200 PRINT "you have the following"
210 PRINT "letters: ";Z$
```



```
220 PRINT "you can type your word now"
230 FOR T=2600 TO 1 STEP -5
240 CALL HCHAR(10,10,INT(T/100)+97)
250 CALL KEY(J,K,S)
260 IF S=0 THEN 410
270 IF K<>8 THEN 340
280 IF B=0 THEN 410
290 Z%=Z%&SEG$(W%,B,1)
300 CALL HCHAR(J,3+B,31)
310 B=B-1
320 W%=SEG$(W%,1,B)
330 GOTO 410
340 C%=CHR$(K)
350 PO=POS(Z%,C%,1)
360 IF PO=0 THEN 410
370 Z%=SEG$(Z%,1,PO-1)&SEG$(Z%,PO+1,B)
380 W%=W%&C%
390 B=B+1
400 CALL HCHAR(J,3+B,K)
410 NEXT T
420 PRINT "your score is: ";B
430 END
```

# Astrology

This program is based on a study made by the Dutch physicist and astrologer Dr. Ir. J. Van Slooten. He was a research worker at Philips laboratories who spent all his free time on astrology. He developed a theory that the phase of the moon at the time of birth was a very important astrological influence on a person's character. After studying the lives of hundreds of people he concluded, ' . . . that the moon phase forecasts the extent to which the spiritual and emotional life, especially with respect to a person's social environment, will develop and furthermore the role he or she will play in our society . . . '

The diagram shows the cycle of the moon's phases. As there are three phases, waxing, full, and waning, so three types of person can be defined.

*Individualists:* born in the waxing moon phase, they have waxing energy. They like to work on their own, have strong wills, and are not discouraged by physical discomfort.

*Socialists:* born in the full moon phase, they like to live in communities. Key words for these people are 'compare', 'choose' and 'combine'. Cooperation is a dominant factor but there is also rivalry and envy.

*Conservatives:* born in the waning moon phase, they know that the light decreases but will return again. They are careful and sure of themselves, and like to have everything under control.

These three types can be further subdivided to give twelve categories in all:

***Pioneers*** are searchers, always ahead of the crowd. Their strength of purpose can sometimes make them appear stubborn.



***Coordinators*** like to be with other people, but feel that they are 'more equal' than their companions. They are romantic and strongly attracted to family life.



***Realists*** look at the world in a very down-to-earth manner. They like to think that everything can be explained in purely physical terms.

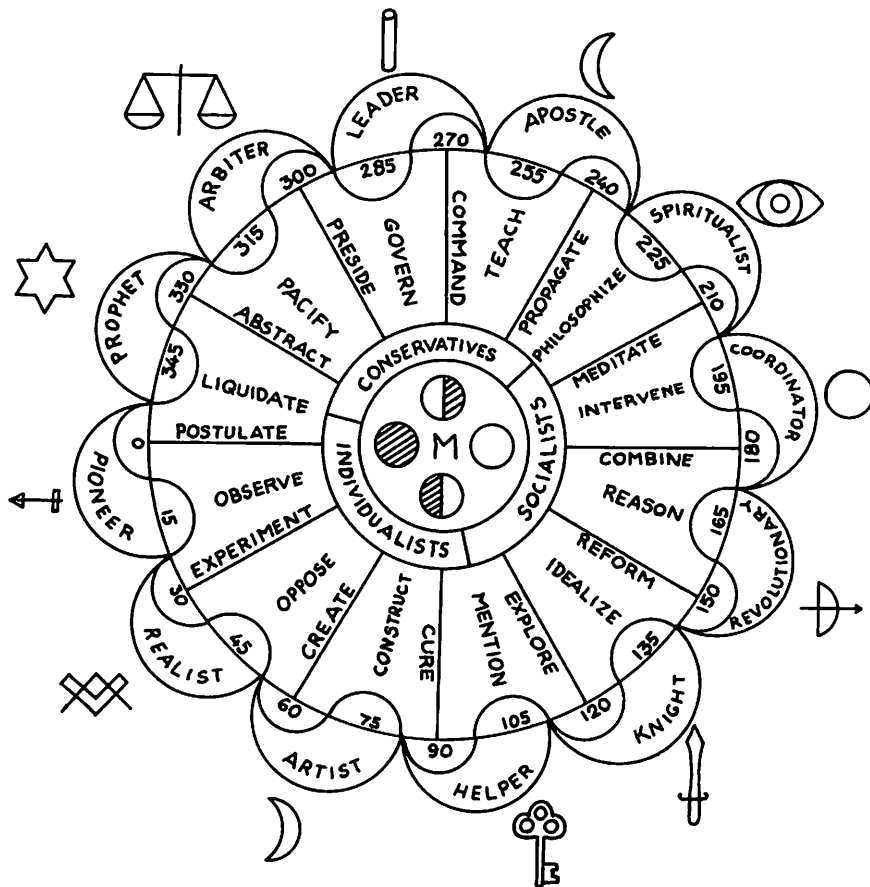


***Spiritualists*** are rather 'other worldly' characters. Philosophical in outlook, they are seekers after the unknown.



***Artists*** transform material in a creative manner. They can produce works of art, new products, or anything that did not exist before.





*Apostles* are not themselves creative but spread the ideas of others to the world at large.



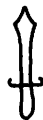
*Helpers* take on responsibilities for their fellows that might otherwise be ignored. They are helpful and caring to others.



*Leaders* also assume responsibilities but in this case for directing the lives and actions of others. They look forward where most people would prefer to look back.



*Knights* are fearless adventurers, always searching for new things to explore. Idealistic in outlook, they have a strong sense of honor.



***Arbiters*** like to preside over the actions of others. They do not prejudge issues, but when they reach a decision they expect it to be obeyed.



***Revolutionaries*** are dissatisfied with the world as it is. They tend to look on the bad side of things, and want to change the world.



***Prophets*** consider the world 'from above' bringing to bear the wisdom of the past. They may not always be listened to, of course.



The program will tell you the phase of the moon on the day you were born. From this you can discover the secrets of your character!



```

10 REM TI-MOONPHASE ASTROLOGY
20 GOTO 290
30 REM input date*****
40 CALL CLEAR
50 PRINT "moonphase"
60 PRINT
70 INPUT "day dd: ":DD
80 INPUT "month mm: ":MM
90 INPUT "year yyyy: ":YYYY
100 M=M-12*(M<3)
110 Y=Y+(M>12)
120 RETURN
130 REM calculation*****
140 T=INT(365.25*YYYY)+INT(30.6*(MM+1))+DD-694038
150 T=T/36525
160 LA=350.737486+1236*T*360
170 LA=LA+307*T+6*T/60
180 LA=LA+51.18*T/3600-5.17*T*1/3600
190 LA=LA-INT(LA/360)*360
200 LA=INT(LA+.5)
210 RETURN
220 REM output*****
230 PRINT
240 PRINT "moonphase=";LA;"degrees"
250 PRINT "press any key"
260 CALL KEY(5,K,S)
270 IF S=0 THEN 260
280 RETURN
290 REM main program*****
300 GOSUB 30
310 GOSUB 130
320 GOSUB 220
330 GOTO 300
340 END

```

> > \$5.95 FPT  
USA

Does your brain get enough exercise? Are your wits as sharp, or your reactions as quick, as they should be? Tone up the muscles of your mind with this enthralling collection of arcade and action games, puzzles, brainteasers and competitions for your microcomputer.

Battle with GALACTIC MONSTERS and ZOMBIES IN THE SWAMP, compete in a thrilling ROAD RACE, defend yourself against a SHIP'S ATTACK, and set out on a nerve-jangling SHARK HUNT.

If you survive all that you can gamble at LAS VEGAS A GO GO and the KENTUCKY DERBY, plumb the mysteries of the BLACK BOX, and stretch your mind with intriguing board games like ONE TO FIVE, SHAKESPEARIAN SHUFFLE, and MINI MANCALA. Then, test your intellectual capabilities with VOWELS AND CONSONANTS, KEYBOARD MEMORY, and AT THE MARKET. Will you turn out to be a GENIUS AT WORK?

By now you'll be razor sharp and ready for anything, which is just as well because you are not even halfway through yet! There's even more excitement, skill and fun for you, your family, and friends. So test your reflexes and mental prowess on this extraordinary obstacle course.

What are you waiting for . . . on your marks, get set . . . GO!